MC68824UM/AD

MC68824
Token Bus Controller User's Manual

MOTOROLA

This document contains information on a new product. Specifications and information herein are subject to change without notice. Motorola reserves the right to make changes to any products herein to improve functioning or design. Although the information in this document has been carefully reviewed and is believed to be reliable, Motorola does not assume any liability arising out of the application or use of any product or circuit described herein; neither does it convey any license under its patent right nor the rights of others.

Motorola, Inc. general policy does not recommend the use of its components in life support applications wherein a failure or malfunction of the component may directly threaten life or injury. Per Motorola Terms and Conditions of Sale, the user of Motorola components in life support applications assumes all risk of such use and indemnifies Motorola against all damages.

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

Section 5
Signals

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Continued)

TABLE OF CONTENTS (Concluded)

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS (Continued)

LIST OF TABLES

SECTION 1
INTRODUCTION

The Motorola MC68824 Token Bus Controller (TBC) is a silicon integrated circuit implementing the media access control (MAC) portion of the IEEE 802.4 token passing bus standard. IEEE 802.4 defines the physical and MAC portion of the data link layer standards of the Manufacturing Automation Protocol (MAP) specification.

The TBC simplifies interfacing a microcomputer to a MAP network by providing the link layer services including managing ordered access to the token bus medium, providing a means for admission and deletion of stations, and handling fault recovery. Some extra features have been added to enhance the token bus controller for real time applications. These enhancements consist of the four levels of priority for transmit and receive and the request with response mechanism. These features, combined with the basic functionality of the token bus controller, make it ideally suited for both seven layer MAP networks and the Enhanced Performance Architecture (EPA) networks defined by MAP version 3.0.

The TBC functions as an intelligent peripheral device to a microprocessor. An on-chip DMA transfers data frames to and from a buffer memory with minimal microprocessor interface required. A microcoded fully linked buffer management scheme queues frames during transmission and reception, and optimizes memory use. This VLSI implementation significantly reduces the cost of a MAP network.

1.1 FEATURES

The MC68824 provides the following:

- Low Power Consumption through 2 Micron HCMOS Fabrication
- MAC Options Suitable for Real Time Environments
  - Four Receive and Four Transmit Queues Supporting Four Priority Levels
  - Immediate Response Mechanism using the Request with Response (RWR) Frame Type
- On-chip Network Monitoring and Diagnostics
- Two Separate Ways of Bridging — Hierarchial and IBM Defined Source Routing
- Powerful Addressing — Group Address Recognition and Multidrop Capability
- System Clock Rate up to 16.67 MHz
- Serial Data Rates from 10 Kbits/Second to 16.67 Mbits/Second
- IEEE 802.4 Recommended Serial Interface Supporting Various Physical Layers
- Simple Interface to Higher Level Software by Means of a Powerful, Fully-Linked Data Structure
- Highly Integrated M68000 Family Bus Master/Slave Interface
  - Four Channel DMA for Transfer of Data Frames to and from Memory
  - 40-Byte FIFO to Efficiently Support High Data Rate
  - 32-Bit Address Bus with Virtual Address Capabilities
- Simplified Interface to Other Processor Environments
  - Byte Swapping Capability for Alternate Memory Structures
  - 8- or 16-Bit Data Bus

1.2 LOCAL AREA NETWORK STANDARDS

Until recently, proprietary local area networks were the only way to tie equipment together to share information and resources. These proprietary local area networks allowed the user to connect small segments of one vendor's computers together. This was all the user needed or wanted. Today, with the increasing complexity of factories and offices, there is a significant increase in the requirements for local area network functionality. Users need to connect more than just a small segment of one vendor's equipment. Since each vendor has its own proprietary protocol, connecting equipment together from more than one vendor or placing a different vendor's machine into a proprietary network can cost up to 80% of the price of the actual equipment. This cost is mainly attributed to the time necessary to write special software programs.

Users began to see that this situation left them with two choices; to buy everything from one vendor, or to buy standard equipment. Since it is obviously not reasonable to buy everything from one vendor, local area network standards began to receive attention. General Motors has specifically addressed the standardization of the factory local area network through the Manufacturing Automation Protocol (MAP) specification based on the ISO/OSI reference model.

1.2.1 ISO/OSI Reference Model

The industry accepted communications model is the International Standards Organization (ISO) reference model of Open Systems Interconnection (OSI). The model specifies seven layers, with each layer responsible for providing specific services to the layer above it. These layers are independent of each other and peer-to-peer layer interfaces are defined and standardized (see Figure 1-1).

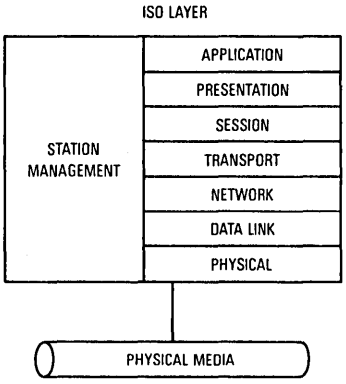


Figure 1-1. IOS/OSI Model

Application Layer
: Provides the network services necessary for the user to run his application program.

Presentation Layer
: Provides the services necessary to format data exchange and to manage session dialog.

Session Layer
: Handles connection establishment, connection termination, and arbitrates session user rights to services.

Transport Layer
: Provides the functions necessary for error free delivery of messages (flow control, acknowledgement, error recovery).

Network Layer
: Routes frames between multiple network segments.

Data Link Layer
: Is responsible for sending frames across the physical link in a reliable manner.

Physical Layer
: Transmits data bits across the physical medium.

1.2.2 IEEE Local Area Network Standards

The IEEE 802 standards body has specified a set of standards based on the lowest two layers of the ISO/OSI model. In order to meet the diverse requirements of the user community, the 802 standards body has defined five different standards so far. These standards are:

1. IEEE 802.3 Carrier Sense Multiple Access with Collision Detection (CSMA/CD) on a Bus Topology
2. IEEE 802.4 Token Bus
3. IEEE 802.5 Token Ring
4. IEEE 802.6 Metropolitan Area Network (MAN)
5. IEEE 802.9 Intergrated Voice Data LAN Access Method

The IEEE standards do not directly correspond to the ISO/OSI model as shown in Figure 1-1. IEEE breaks the data link layer into the media access control (MAC) and the logical link control (LLC) sublayers (see Figure 1-2). The MAC layers are specified by working groups 802.3, 802.4, 802.5, 802.6, and 802.9. The LLC sublayer is specified by 802.2. IEEE has other supporting working groups which consist of network management and interworking (802.1), broadband advisory (802.7), and fiber optic advisory (802.8).

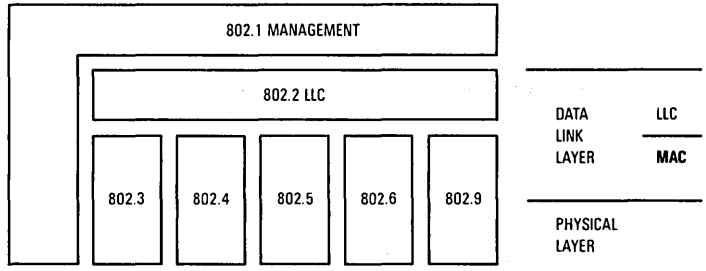


Figure 1-2. IEEE Standard Model

1.3 IEEE 802.4 TOKEN BUS LOCAL AREA NETWORKS

The following paragraphs briefly describe the token bus operation and the options outlined in the IEEE 802.4 standard.

1.3.1 Token Bus Operation

The IEEE 802.4 token passing bus standard is a deterministic protocol. This means that the user is guaranteed access to the network on or before certain predefined intervals. The predefined intervals are dependent on network characteristics such as number of stations and token hold times within each station. A station is only allowed to transmit onto the network when it holds the token. A token is an 8-bit pattern in the frame control portion of the frame. There is only one token on a network. This token is passed from the station with the highest address to the station having the next highest address and so on, with the lowest addressed station passing the token back to the highest addressed station. This circular passing of the token forms a logical ring.

The initialization of a token bus network is accomplished by each station listening to the network. When a particular station hears nothing for a specified period, it will try to claim the token. A number of stations can try to claim the token at the same time but only the highest addressed station will win the token.

Periodically, each station on the network checks to see if a station that has an address between that of itself and the station it passes the token to (its successor in the logical ring), wants to enter the network. If there is such a station, the original station makes the new station the recipient of the token (its successor), therefore allowing the new station to enter the ring. It is conceivable that more than one station could try to enter the logical ring at the same time (i.e., have addresses between the same two stations). The protocol is set up such that the highest addressed station will be allowed into the ring and the other station(s) will have to try again at the next opportunity.

When a station no longer desires to be a part of the logical ring, it tells the station preceeding it (previous station) that it is leaving the ring. The previous station will then determine its new successor, and pass it the token.

For more detailed information on how the IEEE 802.4 standard works, refer to **APPENDIX A IEEE 802.4 OPERATION** or the IEEE 802.4 standard.

1.3.2 Options Within IEEE 802.4

The token bus controller implements both the priority and request with response (RWR) options outlined in the IEEE 802.4 standard. These options make the TBC suitable for use in real time networks as specified in MAP version 3.0.

The request with response option allows any station that is a member of the logical ring to communicate with any station(s) that is not a member of the logical ring. This is useful in real time networks because stations that do not have much data to transmit do not add unnecessary delay to the token rotation time. The longer the token rotation time is, the longer it is between each station's opportunity to transmit. Not being a part of the logical ring allows these stations to have less functionality, simpler software, and lower cost. Whenever a station needs data from a station that is not a member of the logical ring, the requesting station sends a request with response frame to the station it wants a response from. The responding station then returns a response while the requesting station still holds the token.

The four levels of priority are optional. If the user chooses not to implement priority, all frames are received and transmitted out of the highest priority queue. If priority is implemented, frames are linked into one of the four queues according to their priority. The TBC implements priority by transmitting out of the highest priority queue until there are no more frames in that queue or until the timer for that priority has elapsed. The timer for the highest priority queue is set by the user while the timer for lower priorities is set partially by the user and partially by network history. The TBC then either passes the token or checks the next lower priority queue for frames to transmit. If the timer for the next lower queue has not elapsed (the TBC has not passed the token), the TBC transmits out of that queue until there are no more frames to transmit or until the timer for that priority (which has been set by the user) has elapsed, and so on through all four queues. Finally the TBC passes the token to the next station. While receiving, frames of the same priority are linked together so the host can easily read the data by priority queue.

1.4 TBC ENVIRONMENT

The token bus controller provides ordered access to the token bus medium while under the overall supervision of a microprocessor. The communication between the microprocessor and the TBC is primarily through shared memory, but also uses a command structure.

Figure 1-3 illustrates a typical system configuration using the TBC. The TBC, memory, and CPU all communicate through a local bus. On the serial side, the TBC communicates through the IEEE 802.4G recommended standard serial interface to any modem also implementing this interface.

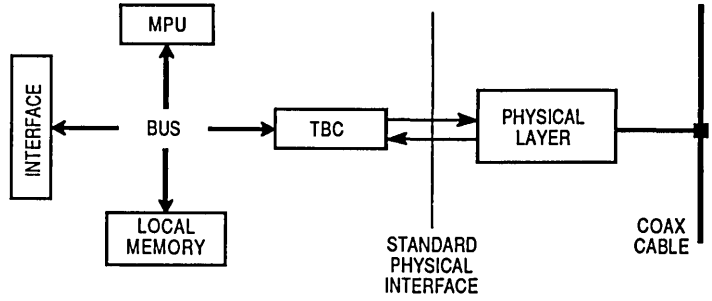


Figure 1-3. Token Bus LAN Node

1.5 PHYSICAL LAYER INTERFACE

The TBC is a half-duplex controller which can be interfaced to any physical layer that implements the IEEE 802.4 recommended standard serial interface at 1, 5, or 10 Mbps. The serial interface supports two modes of operation: station management mode and MAC mode. In station management mode, commands may be given to an intelligent modem through the TBC. The serial interface consists of a request channel and an indication channel. The request channel is from the TBC to the modem and consists of TXCLK (supplied by the modem), TXSYM0, TXSYM1, TXSYM2, and $\overline{\text{SMREQ}}$. The indication channel is from the modem to the TBC and consists of RXCLK (supplied by the modem), RXSYM0, RXSYM1, RXSYM2, and $\overline{\text{SMIND}}$.

Both the TBC and the modem can request station management mode. The TBC requests station management mode in order to give commands and the modem requests station management mode in order to report error conditions. The TBC can either encode commands on its TXSYM lines or send serial data commands on one of the TXSYM lines.

1.6 MICROPROCESSOR SYSTEM BUS INTERFACE

The host microprocessor can be any 8-, 16-, or 32-bit microprocessor. The microprocessor interface on the TBC is that of the M68000 Family. The TBC provides byte swapping capability to enable the user to operate with either Motorola or Intel byte ordering convention.

The TBC has a 32-bit non-multiplexed address bus and a separate 16-bit data bus, which may be configured for 8-bit operation.

1.7 SHARED MEMORY

The TBC and the host microprocessor interface through shared memory which consists of:

- Initialization Table
- Private Area
- Fully Linked Buffer Structure
  - Frame Descriptors
  - Buffer Descriptors
  - Data Buffers

Initialization Table

Contains all of the initial parameters necessary for TBC network operation and is prepared by the host prior to the initialization of the TBC.

Private Area

An external scratchpad for the TBC containing some TBC parameters.

Fully Linked Buffer Structure

Consists of frame descriptors, buffer descriptors, and data buffers.

Frame descriptors contain control information for each frame and are linked together to form each priority queue.

Buffer descriptors are pointed to by frame descriptors and contain control information for the data. Buffer descriptors contain an offset value set by the user which tells how far from the beginning of the data buffer the actual data begins. The offset feature can be used to add or subtract upper layer headers making it unnecessary to recopy data.

Data buffers contain only data and are flexible in size.

1.8 OVERVIEW OF TBC FUNCTIONAL OPERATION

The following paragraphs provide a high level summary of the command structure, frame reception, and frame transmission.

1.8.1 COMMAND STRUCTURE OVERVIEW

The host issues a command to the TBC by writing the 8-bit command into the command register. The commands belong to one of the following categories:

- Initialization
- Set Operation Mode
- Transmit Data Frames
- Set/Read Value
- Test
- Notify TBC
- Modem Control

1.8.2 FRAME RECEPTION OVERVIEW

A frame is received by a station if the address comparison is true. For group addressed frame reception, the address comparison uses the group address mask, this station's address, and the

destination address of the frame. The address comparison for a non-group (individual) addressed frame uses the individual address mask, this station's address, and the destination address of the frame. Broadcast frames are always accepted. Once the station determines that it is to receive the frame, the frame is transferred by DMA into the linked buffer portion of shared memory into the appropriate priority queue. The TBC does all of the linking and transferring. After the frame has been written into shared memory, an indication of its reception and its status are written into the frame descriptor.

1.8.3 FRAME TRANSMISSION OVERVIEW

Data frames are transmitted from one of four transmission priority queues. Frames are only transmitted out of the enabled priority queues. On reception of the token by the station, the TBC checks the transmission queues for frames to be sent. If frames are present on one or more of the queues, the TBC will send the frames until its allotted transmission time has expired, then pass the token. Confirmation that a specific frame has been sent, along with the transmission status is written into its frame descriptor.

SECTION 2
TABLES

The MC68824 Token Bus Controller (TBC) utilizes two tables to interface with the host. The tables include the TBC private area and the initialization table. The TBC private area is a 128-byte block of memory used by the TBC to store internal variables and pointers. The initialization table is a 256-byte block of memory which holds initial parameters, interrupt status and interrupt masks, a DMA dump area, statistical information, and the command parameter area (CPA). The CPA is used by the host to set and read TBC parameters.

2.1 TBC PRIVATE AREA

The TBC private area is a 128-byte area of RAM reserved for use by the TBC to store internal variables and statistical information associated with the media access control (MAC) operation. During initialization, the host CPU specifies the appropriate initial values of the private area parameters in the initialization table (see displacement 08 hex through 76 hex in the initialization table shown in Figure 2-3). These initial values are then loaded into the private area through the INITIALIZE command (see **3.2.2 INITIALIZE Command** for details). After initialization, the TBC keeps an on-chip pointer to this private area. The private area should never be directly accessed by the host as this would not guarantee IEEE 802.4 operation. The parameters in the private area should only be set and read by the SET/READ VALUE commands through the command parameter area (see **3.5 SET/READ VALUE**).

The format of the TBC private area is shown in Figure 2-2. Those parameters specified in the IEEE 802.4 standard are noted. All timers are specified in octet times; one octet time is defined as 8•1/ (network data rate)). The data is organized following the Motorola byte ordering convention unless otherwise specified. That is, the low-order byte has an odd address while the high-order byte has an even address as shown in Figure 2-1.

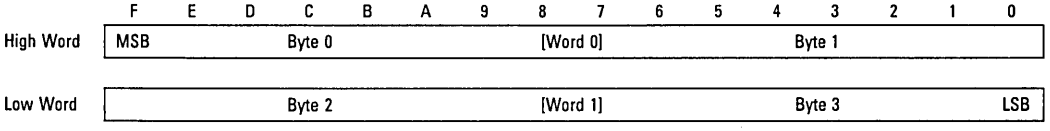


Figure 2-1. Data Organization in Memory

2.1.1 Next Station Address

Next station (NS) address is the 48-bit (or 16-bit) address of the next station in the logical ring that this station will pass the token to. The address length of 48 or 16 bits for the station is chosen at initialization time by setting or resetting a bit located in offset 7A of the initialization table. The

DISPLACEMENT (IN BYTES)	DESCRIPTION OF FIELD	
00	NEXT STATION ADDRESS HIGH	802.4
02	NEXT STATION ADDRESS MEDIUM	802.4
04	PREVIOUS STATION ADDRESS HIGH	802.4
06	PREVIOUS STATION ADDRESS MEDIUM	802.4
08	HI_PRIORITY_TOKEN_HOLD_TIME	802.4
0A	LAST TOKEN_ROTATION_TIME	802.4
0C	TOKEN ROTATION TIME AT START OF ACCESS CLASS 4	
0E	TARGET_ROTATION_TIME FOR ACCESS CLASS 4	802.4
10	TOKEN ROTATION TIME AT START OF ACCESS CLASS 2	
12	TARGET_ROTATION_TIME FOR ACCESS CLASS 2	802.4
14	TOKEN ROTATION TIME AT START OF ACCESS CLASS 0	
16	TARGET_ROTATION_TIME FOR ACCESS CLASS 0	802.4
18	TOKEN ROTATION TIME AT START OF RING MAINTENANCE	
1A	TARGET_ROTATION_TIME FOR RING MAINTENANCE	802.4
1C	RING MAINTENANCE TIME INITIAL VALUE	802.4
1E	SOURCE ROUTING — SOURCE SEGMENT/BRIDGE ID (SID)	
20	SOURCE ROUTING — TARGET SEGMENT/BRIDGE ID (TID)	
22	SEGMENT NUMBER MASK FOR SOURCE ROUTING	
24	MAX_INTER_SOLICIT_COUNT	802.4
26	RX FRAME STATUS ERROR MASK	
28	TX QUEUE ACCESS CLASS 6 STATUS	
2A	TX QUEUE ACCESS CLASS 6 HEAD OF QUEUE POINTER	
2E	ZERO	
30	TX QUEUE ACCESS CLASS 4 STATUS	
32	TX QUEUE ACCESS CLASS 4 HEAD OF QUEUE POINTER	
34	ZERO	
36	TX QUEUE ACCESS CLASS 2 STATUS	
38	TX QUEUE ACCESS CLASS 2 HEAD OF QUEUE POINTER	
3A	ZERO	
3E	ZERO	
40	TX QUEUE ACCESS CLASS 0 STATUS	
42	TX QUEUE ACCESS CLASS 0 HEAD OF QUEUE POINTER	
46	ZERO	
48	RX QUEUE ACCESS CLASS 6 END OF QUEUE POINTER	
4C	RX QUEUE ACCESS CLASS 4 END OF QUEUE POINTER	
50	RX QUEUE ACCESS CLASS 2 END OF QUEUE POINTER	
54	RX QUEUE ACCESS CLASS 0 END OF QUEUE POINTER	
58	FREE FRAME DESCRIPTOR POINTER TO FIRST FD	
5C	FREE BUFFER DESCRIPTOR POINTER TO FIRST BD	
60	GROUP ADDRESS MASK — LOW	
62	GROUP ADDRESS MASK — MEDIUM	
64	GROUP ADDRESS MASK — HIGH	
66	INDIVIDUAL ADDRESS MASK — LOW	
68	INDIVIDUAL ADDRESS MASK — MEDIUM	
6A	INDIVIDUAL ADDRESS MASK — HIGH	
6C	NON RWR MAXIMUM RETRY LIMIT	
6E	RWR MAXIMUM RETRY LIMIT	802.4
70	SLOT TIME	802.4
72	THIS STATION ADDRESS — LOW	802.4
74	THIS STATION ADDRESS — MEDIUM	802.4
76	THIS STATION ADDRESS — HIGH	802.4
78	MANUFACTURER PRODUCT VERSION AND 802.4 STANDARD REV. NUMBER	
7A	TRANSMITTER FAULT COUNT	802.4
7C	RESERVED FOR TBC	
7E	RESERVED FOR TBC	

Figure 2-2. TBC Private Area

low word of this address is stored in an internal TBC register while the middle and high words are stored in the private area. Next station address may be accessed through the READ VALUE command (see **3.5.2 READ VALUE** for details).

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
00	MSB								Next Station — High Word							
02									Next Station — Middle Word							LSB

2.1.2 Previous Station Address

Previous station (PS) address is the 48-bit (or 16-bit) address of the previous station that passes the token to this station. The low word of this address is stored in an internal TBC register with the middle and high words stored in the private area. Previous station address is accessed through the READ VALUE command.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
04	MSB								Previous Station — High Word							
06									Previous Station — Middle Word							LSB

2.1.3 Hi_Piority_Token_Hold Time

The hi_priority_token_hold time can be from 0 to (2^{16})-1 octet times before prescaling (see **3.3.3 SET MODE 3** for details on scaling). If the priority option is used, the hi_priority_token_hold_time is the maximum amount of time the station may transmit data from the highest priority queue (queue 6). If the priority option is not used, the hi_priority_token_hold_time determines the maximum amount of time the station can transmit before passing the token. The IEEE 802.4 standard specifies the maximum value for this timer to be (2^{16})-1 octet times which therefore requires the user to zero the upper three bits or the upper six bits of this word depending on the prescaling factor used. However, the TBC does not check that this parameter is within the specified range which allows the user to specify a timer with a maximum value of (2^{22})-1. This parameter must be initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
08	0	0	0	Hi_Priority_Token_Hold Time (Prescaler of 3)												
08	0	0	0	0	0	0	Hi_Priority_Token_Hold Time (Prescaler of 6)									

2.1.4 Last Token_Rotation_Time

Last token_rotation_time is the observed prescaled-octet time measured from the last time the station had the token to when the station receives the token again, timed from token arrival to token arrival. This statistic is updated every time the TBC receives the token. Last token_rotation_time can be accessed by the host via the READ VALUE command. The worst case token_rotation_time is equal to the total of the token hold times for all the nodes on the network plus the time it takes to pass the token between nodes. A value of zero indicates that the value is not currently available and occurs, for example, when the token is not rotating.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
0A	Last Token_Rotation_Time (Prescaled)															

2.1.5 Token Rotation Time at Start of Access Classes

Token rotation time at the start of each access class is measured in octet times from the station's receipt of the token to the TBC's entrance into that access class. There is one token rotation timer for each of the lower three access classes and one for ring maintenance. These statistics along with the target rotation time for each access class are used by the TBC to implement the priority mechanism. These statistics have no meaning if the priority access class is disabled or if the station is not a member of the logical ring. These statistics can be accessed by the host via the READ VALUE command.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
0C, 10, 14	Token Rotation Time at Start of Access Class (Prescaled)															

2.1.6 Target Token Rotation Time for Access Classes

The target token rotation time for each access class is user programmable in the range from 0 to (2^{21})-1 octet times before prescaling, and is used in conjunction with the token rotation time at start of access classes and ring maintenance timer. These parameters must be initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter. The TBC does not check that these parameters are within the specified range. There is a set of four timers called token rotation timers which get loaded with the target token rotation time as explained below. There is one token rotation timer for each of the lower three access classes and one for ring maintenance. These timers all run concurrently, counting downward from an initial value to zero, at which point their status is expired. When the station begins processing the token at a given access class, the associated internal token rotation timer is reloaded with the value of the target token rotation time for that access class. When the station receives the token again it may send data from that access class until the residual time in the associated token rotation timer has expired.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
0E, 12, 16,	0	Target Rotation Time for Access Classes (Prescaled)														

2.1.7 Token Rotation Time at Start of Ring Maintenance

The token rotation time at the start of ring maintenance is a statistic which contains the elapsed time in units of prescaled octet times from arrival of the token to entering ring maintenance. This statistic can be accessed by the host via the READ VALUE command.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
18	Token Rotation Time at Start of Ring Maintenance (Prescaled)															

2.1.8 Target Rotation Time for Ring Maintenance

The target rotation time for ring maintenance is initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter. This parameter gets loaded into the internal ring maintenance token rotation timer and is in the range from 0 to $(2^{21})-1$ octet times before prescaling. The station will solicit a successor if the inter_solicit_count equals zero and the ring maintenance token rotation timer has not expired. If the timer has expired, the solicitation will be deferred until the next time the station holds the token.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
1A	0	Target Rotation Time for Ring Maintenance (Prescaled)														

2.1.9 Ring Maintenance Time Initial Value

The ring maintenance time initial value is an integer in the range from 0 to $(2^{21})-1$ octet times before prescaling. This parameter must be initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter. This initial value gets loaded into the ring maintenance timer upon initial entry into the ring. If the ring maintenance initial value is set to zero, the station will defer solicitation of successors for at least one rotation of the token. If the value is high, the station will immediately solicit successors upon entry to the logical ring.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
1C	0	Ring Maintenance Time Initial Value (Prescaled)														

2.1.10 Source Routing — Source Segment/Bridge ID (SID)

Two 16-bit segment numbers (SID and TID) are required by the optional source routing mechanism as described in **APPENDIX C BRIDGING**. These parameters determine whether or not the TBC accepts a particular source routing frame.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
1E	Source Routing Source ID															

2.1.11 Source Routing — Target Segment/Bridge ID (TID)

The target number for source routing is discussed in the **APPENDIX C BRIDGING.**

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
20	Source Routing Target ID															

2.1.12 Segment Number Mask for Source Routing

The segment number mask determines which part of the routing designator is the segment number and which part is the bridge number. See **APPENDIX C BRIDING** for details.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
22	Segment Number Mask for Source Routing															

2.1.13 Max_Inter_Solicit_Count

Max_Inter_Solicit_Count, working with the ring maintenance timer and target rotation timer for ring maintenance, determines how often the station opens a response window. The range of this parameter should be from 16 to 255, however the TBC will accept values smaller than 16. As required by the 802.4 standard, the TBC ignores the two least significant bits of the given value and replaces them by two random bits. If the inter_solicit_count equals zero and if the ring maintenance token rotation timer has not expired, the station will open a window to solicit a new successor. If the ring maintenance timer has expired, the solicitation will be deferred until the next time the station holds the token. This parameter must be initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
24	Max_Inter_Solicit_Count								0	0	0	0	0	0	0	0

2.1.14 RX Frame Status Error Mask

If a frame addressed to the TBC is received with one of the errors listed below and the corresponding error mask bit is set to one, the TBC will accept the frame (for more details see **4.1.1.2 RECEIVE STATUS WORD**). The format of the status error mask is shown below. This parameter is initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
26	CRCE	EBIT	FOVR	NOISE	FTL	NOBUF	0	0	0	0	0	0	0	0	0	0

CRCE CRC Error
EBIT E-Bit Error
FOVR FIFO Overrun
NOISE Noise
FTL Frame Too Long (>8 K)
NOBUF Not Enough Buffers in Free Buffer Pool

2.1.15 TX Queue Access Class Status

Each transmit queue has an associated transmit queue access class status word. Each status word is initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter. In each of the access class status words, bit F (QD) contains the queue disabled bit and bit E (QE) contains the queue empty bit. In order to transmit, the host processor may set the queue disabled bit to zero to enable the queue and the empty bit to zero indicating the queue contains data. This action is not necessary since the START command sets both bits (QD and QE) of the status word to zero, indicating that the queue is enabled and full. When the TBC is finished transmitting from the queue, it sets the queue empty bit to one. A transmit queue, except for class 6 frames, must be enabled at least one token rotation time before it can contain data or before the station enters the logical ring. This is done so the token rotation timers for each access class have a valid value.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
28, 30, 38, 40	QD	QE	0	0	0	0	0	0	0	0	0	0	0	0	0	0

QD — Queue Disabled
1 Queue is Disabled
0 Queue is Enabled

QE — Queue Empty
1 Queue is Empty
0 Queue is Not Empty

2.1.16 TX Queue Access Class Pointers

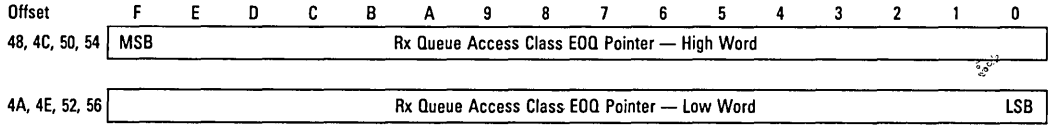
The four transmit queue access class pointers point to the first frame descriptor in the respective access class queue. These pointers direct the TBC to the location from which to start transmitting. Each pointer is initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter. For more details on the management of these pointers refer to **SECTION 4 BUFFER STRUCTURES**. The format is shown below.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
2A, 32, 3A, 42	MSB	Tx Queue Access Class HOQ Pointer — High Word														
2C, 34, 3C, 44	Tx Queue Access Class HOQ Pointer — Low Word															LSB

2.1.17 RX Queue Access Class Pointers

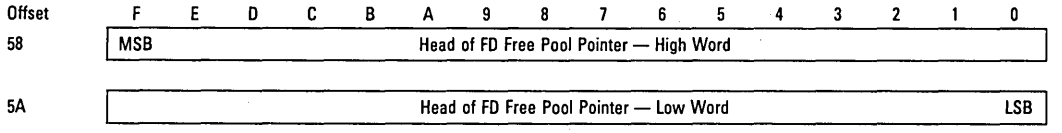
The four receive queue access class pointers point to the last frame descriptor in the respective access class queue. Each pointer is initialized by the host through the initialization table and may

be altered or read via the SET/READ VALUE command thereafter. This enables the TBC to add onto the linked list. Initially these receive queue access class pointers must point to a valid frame descriptor for the TBC to link from. This frame descriptor, referred to as a 'dummy' frame descriptor, must always be provided. For more details on the management of these pointers refer to **SECTION 4 BUFFER STRUCTURES.**



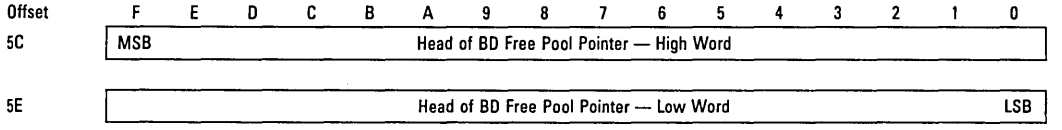
2.1.18 Free Frame Descriptor Pointer

The free frame descriptor pointer is a 32-bit pointer containing the address of the first free frame descriptor in the linked list of the free frame descriptor pool. This pointer is initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter. The TBC uses these free frame descriptors upon reception of a frame. Initially, the free frame descriptor pointer must point to a valid frame descriptor. For more details on the management of these pointers refer to **SECTION 4 BUFFER STRUCTURES.**



2.1.19 Free Buffer Descriptor Pointer

The free buffer descriptor pointer is a 32-bit pointer containing the address of the first free buffer descriptor in the linked list of the free buffer descriptor pool. The TBC uses these free buffer descriptors upon reception of a frame. This pointer is initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter. For more details on the management of these pointers refer to **SECTION 4 BUFFER STRUCTURES.**

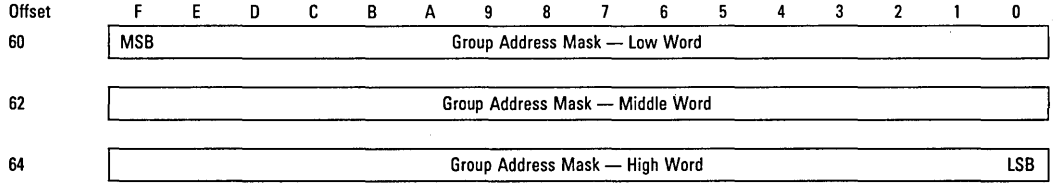


2.1.20 Group Address Mask

The group address mask is used to determine if a group addressed frame is to be accepted by the station. A group addressed frame is denoted by the least significant bit of the destination address field. This bit, called the I/G bit, must be one for a group addressed frame. A group address is used to send messages to a group of logically related stations. The group address mask is either two or six bytes in length, depending on the addressing mode of the TBC, and must have its least significant bit set to one. The group address mechanism can be disabled by setting the group address mask to zero. Broadcast frames will always be accepted, regardless of the group address mask. The mathematical expression for accepting frames using the group address mechanism is:

Destination Address AND GA_MASK = Destination Address

If the boolean value of the expression is true, then the frame is accepted. For a more detailed explanation and examples of group addressing, see **APPENDIX B FRAME FORMATS AND ADDRESSING**. This parameter is initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter. The middle and high words must be zero if the address length is two bytes. Note that the group address mask is stored following the IEEE format.

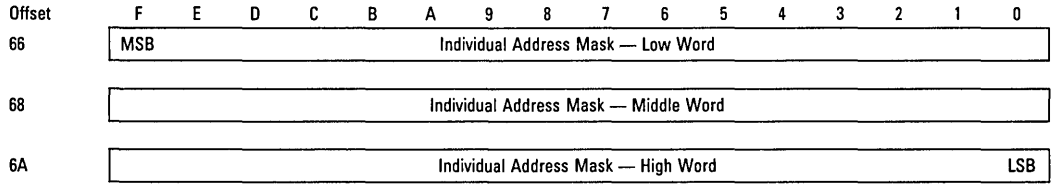


2.1.21 Individual Address Mask

The individual address mask can be either two or six bytes and must have its least significant bit set to zero. It is used to modify the test used by the TBC for individual node address recognition. The mathematical expression for accepting frames using the individual address mechanism is:

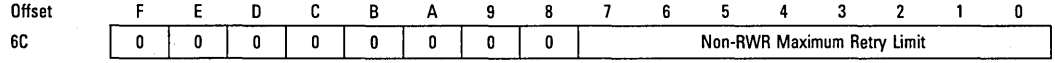
Destination Address AND IA_MASK = This Station Address AND IA_MASK

If the boolean value of the expression is true, then the frame is accepted. For a more detailed explanation and examples of individual addressing, see **APPENDIX B FRAME FORMATS AND ADDRESSING**. This parameter is initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter. The middle and high words should be zero if the address length is 16 bits. Note that the individual address mask is stored following the IEEE format.



2.1.22 Non-RWR Maximum Retry Limit

The maximum number of non-request with response retries sets an upper bound to the number of times the TBC will attempt to transmit the same message. This parameter is initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter. Trying to send a frame again (retrying) is necessary when long host bus latency conditions occur, sometimes causing the frame transmission to be aborted. The TBC will keep attempting to send a message until this retry limit is met. This maximum retry limit is user programmable with the TBC supporting a maximum of 255 retries. If the frame still has not been sent when the maximum retry limit is met, negative confirmation appears in the confirmation word of the frame descriptor and the TBC will stop trying to send it. The maximum retry limit for non-RWR messages is not an IEEE 802.4 parameter.



2.1.23 RWR Maximum Retry Limit

The request with response (RWR) maximum retry limit sets an upper bound to the number of times the TBC will attempt to transmit a RWR frame. A retry is performed if an underrun occurs or if no response is received from the destination address. This parameter is initialized by the host through the initialization table and may be altered or read via the SET/READ VALUE command thereafter. In the IEEE 802.4 standard, the maximum RWR number of retries value is seven. The maximum value that the TBC will support is 255.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
6E	0	0	0	0	0	0	0	0	RWR Maximum Retry Limit							

2.1.24 Slot Time

Slot time is the worst case time any station must wait for a response from another station. This parameter must be uniform throughout the network. It is equal to the amount of time for the slowest station on the bus to respond to MAC control frames. Slot time is measured in octet times, has a maximum value of $(2^{13})-1$ and is defined by the equation below:

$$\text{SLOT TIME} = \text{Integer}\ ((2 \cdot \text{Transmission\_Path\_Delay}) + (((2 \cdot \text{Station\_Delay}) + \text{Safety\_Margin})/\text{MAC\_Symbol\_Time})/8)$$

The transmission path delay is the worst case delay which transmissions experience going through the physical medium from a transmitting station to a receiving station. The station delay is the amount of time from the receipt of the end delimiter at the receiving station's physical medium interface until the impression of the first bit of the preamble on the physical medium by the receiving station's transmittter. The safety margin in the time interval is no less than one MAC symbol time. MAC symbols are defined as the smallest unit of information exchanged between the MAC sublayer entities. MAC symbol time is the time required to send a single MAC symbol and is therefore the inverse of the network data rate.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
70	0	0	0	Slot Time (In Octets)												

2.1.25 This Station Address

This station address (TS) is a 48-bit or 16-bit address which must be an even integer since an odd address represents a group address. It contains the address used by the TBC to accept or reject MAC frames. This parameter is initialized by the host through the initialization table and may be read via the READ VALUE command thereafter. The middle and high words should be zero if the address length is 16 bits. Note that this parameter is stored following the IEEE format.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
72	MSB	This Station Address — Low Word														
74	This Station Address — Middle Word															
76	This Station Address — High Word															LSB

2.1.26 Manufacturer Product Revision and 802.4 Standard Revision Number

The value of this parameter is "0402 hex" for TBC mask set 2A60S and "0502 hex" for TBC mask set 1B59B. It is stored at offset 78. The '4' or '5' is an indication of the manufacturer product

revision while the '2' means the TBC implements 802.4 Revision 2. This parameter may be read via the READ VALUE command.

2.1.27 Transmitter Fault Count

2

The transmitter fault count is an integer in the range from zero to seven stored by the TBC in the most significant three bits. If the station sequences to the end of the token contention process and loses, or fails to pass the token to any successor, this counter is incremented by one. If this value exceeds seven, bit 1 of interrupt status word 1 will be set by the TBC (see **2.2.11 Interrupt Status Words** for a complete description). This parameter may be read via the READ VALUE command.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
7A	TX FCNT			0	0	0	0	0	0	0	0	0	0	0	0	0

2.2 INITIALIZATION TABLE

The initialization table is a 256-byte area of memory which is shared by the TBC and the host CPU. It is used by the host processor to pass operating parameters and pointers to the TBC. This is performed during the initialization sequence and whenever it is necessary to update operating parameters during TBC operation. The initialization table is also used by the TBC to pass status, statistic counters, and command return information (parameters) to the host.

The format of the initialization table is shown in Figure 2-3.

2.2.1 Private Area Function Code

Bits 0-3 of the initialization table word 0 contain the function code values that may be required by the system to access the private area. If no function codes are used, this word does not need to be initialized by the host. The TBC private area pointer is located in the subsequent two words of the initialization table.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
00	0	0	0	0	0	0	0	0	0	0	0	0	Private Area FC			

2.2.2 Private Area Pointer

The private area pointer is a 32-bit address which points to the 128-byte area in RAM that should only be accessed by the TBC since it contains MAC variables and parameters.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
02	MSB	Private Area Pointer — High Word														
04	Private Area Pointer — Low Word															LSB

2.2.3 Parameters Initializing the Private Area

The parameters in displacements 08-76 are the initial values of the corresponding parameters of the private area. These values are initialized by the host CPU in the intialization table and are then loaded into the private area by the INITIALIZE command. For a detailed description of these parameters, see **2.1 TBC PRIVATE AREA.**

DISPLACEMENT (IN BYTES)	DESCRIPTION OF FIELD	UPDATED BY
00	PRIVATE AREA FUNCTION CODE	Host
02	PRIVATE AREA POINTER — HIGH	Host
04	PRIVATE AREA POINTER — LOW	Host
06	ZERO	Host
08	INITIAL HI_PRIORITY_TOKEN_HOLD_TIME	Host
0A	ZERO	Host
0C	ZERO	Host
0E	INITIAL TARGET_ROTATION_TIME FOR ACCESS CLASS 4	Host
10	ZERO	Host
12	INITIAL TARGET_ROTATION_TIME FOR ACCESS CLASS 2	Host
14	ZERO	Host
16	INITIAL TARGET_ROTATION_TIME FOR ACCESS CLASS 0	Host
18	ZERO	Host
1A	INITIAL TARGET_ROTATION_TIME FOR RING MAINTENANCE	Host
1C	INITIAL RING MAINTENANCE TIME INITIAL VALUE	Host
1E	INITIAL SOURCE SEGMENT/BRIDGE ID (SID)	Host
20	INITIAL TARGET SEGMENT/BRIDGE ID (TID)	Host
22	INITIAL SEGMENT NUMBER MASK FOR SOURCE ROUTING	Host
24	INITIAL MAX_INTER_SOLICIT_COUNT	Host
26	INITIAL RX FRAME STATUS ERROR MASK	Host
28	INITIAL TX QUEUE ACCESS CLASS 6 STATUS	Host
2A	INITIAL TX QUEUE ACCESS CLASS 6 HOQ POINTER	Host
2E	ZERO	Host
30	INITIAL TX QUEUE ACCESS CLASS 4 STATUS	Host
32	INITIAL TX QUEUE ACCESS CLASS 4 HOQ POINTER	Host
36	ZERO	Host
38	INITIAL TX QUEUE ACCESS CLASS 2 STATUS	Host
3A	INITIAL TX QUEUE ACCESS CLASS 2 HOQ POINTER	Host
3E	ZERO	Host
40	INITIAL TX QUEUE ACCESS CLASS 0 STATUS	Host
42	INITIAL TX QUEUE ACCESS CLASS 0 HOQ POINTER	Host
46	ZERO	Host
48	INITIAL RX QUEUE ACCESS CLASS 6 EOQ POINTER	Host
4C	INITIAL RX QUEUE ACCESS CLASS 4 EOQ POINTER	Host
50	INITIAL RX QUEUE ACCESS CLASS 2 EOQ POINTER	Host
54	INITIAL RX QUEUE ACCESS CLASS 0 EOQ POINTER	Host
58	INITIAL FREE FRAME DESCRIPTOR POOL POINTER TO FIRST FD	Host
5C	INITIAL FREE BUFFER DESCRIPTOR POOL POINTER TO FIRST BD	Host
60	INITIAL GROUP ADDRESS MASK — LOW	Host
62	INITIAL GROUP ADDRESS MASK — MEDIUM	Host
64	INITIAL GROUP ADDRESS MASK — HIGH	Host
66	INITIAL INDIVIDUAL ADDRESS MASK — LOW	Host
68	INITIAL INDIVIDUAL ADDRESS MASK — MEDIUM	Host
6A	INITIAL INDIVIDUAL ADDRESS MASK — HIGH	Host
6C	INITIAL NON RWR MAX RETRY LIMIT	Host
6E	INITIAL RWR MAX RETRIES LIMIT	Host
70	INITIAL SLOT TIME	Host
72	INITIAL THIS STATION ADDRESS — LOW	Host
74	INITIAL THIS STATION ADDRESS — MEDIUM	Host
76	INITIAL THIS STATION ADDRESS — HIGH	Host
78	INITIAL PAD TIMER PRESET (PTP)	Host
7A	BIT 15 — INITIAL IN_RING DESIRED	Host
	BIT 14 — INITIAL ADDRESS LENGTH (48/16 BITS)	Host
7C	ZERO	Host
7E	ZERO	Host

Figure 2-3. Initialzation Table (Sheet 1 of 2)

DISPLACEMENT (IN BYTES)	DESCRIPTION OF FIELD	UPDATED BY
80	ZERO	Host
82	ZERO	Host
84	RESPONSE DESTINATION ADDRESS POINTER	Host
88	RESPONSE POINTER	TBC
8C	RWR POINTER	
	COMMAND PARAMETER AREA	
90	COMMAND PARAMETER AREA VAL0	Host
92	COMMAND PARAMETER AREA VAL1	Host
94	COMMAND PARAMETER AREA VAL2	Host
96	COMMAND RETURN AREA RET0	TBC
98	COMMAND RETURN AREA RET1	TBC
9A	COMMAND RETURN AREA RET2	TBC
9C	COMMAND STATUS AND DONE BIT	TBC
9E-A6	ZERO	Host
	INTERRUPT STATUS AREA	
A8	INTERRUPT STATUS WORD 0	TBC
AA	INTERRUPT MASK 0	Host
AC	INTERRUPT STATUS WORD 1	TBC
AE	INTERRUPT MASK 1	Host
	STATISTICS	
B0	NUMBER OF TOKENS PASSED (THRESHOLD)	Host
B2	NUMBER OF TOKENS PASSED	TBC
B4	NUMBER OF TOKENS HEARD (THRESHOLD)	Host
B6	NUMBER OF TOKENS HEARD	TBC
B8	NUMBER OF NO_SUCCESSOR_8 ARCS (THRESHOLD)	Host
BA	NUMBER OF NO_SUCCESSOR_8 ARCS	TBC
BC	NUMBER OF WHO_FOLLOWS TRANSMITTED (THRESHOLD)	Host
BE	NUMBER OF WHO_FOLLOWS TRANSMITTED	TBC
C0	NUMBER OF TOKEN PASSES THAT FAILED (THRESHOLD)	Host
C2	NUMBER OF TOKEN PASSED THAT FAILED	TBC
C4	NUMBER OF NON_SILENCE (THRESHOLD)	Host
C6	NUMBER OF NON_SILENCE	TBC
C8	NUMBER OF FCS ERRORS (THRESHOLD)	Host
CA	NUMBER OF FCS ERRORS	TBC
CC	NUMBER OF E-BIT ERRORS (THRESHOLD)	Host
CE	NUMBER OF E-BIT ERRORS	TBC
D0	NUMBER OF FRAME FRAGMENTS (THRESHOLD)	Host
D2	NUMBER OF FRAME FRAGMENTS	TBC
D4	NUMBER OF RECEIVED FRAMES TOO LONG (>8K BYTES) (THRESHOLD)	Host
D6	NUMBER OF RECEIVED FRAMES TOO LONG	TBC
D8	NUMBER OF NO FD/BD ERRORS (THRESHOLD)	Host
DA	NUMBER OF NO FD/BD ERRORS	TBC
DC	NUMBER OF OVERRUNS (THRESHOLD)	Host
DE	NUMBER OF OVERRUNS	TBC
	DMA DUMP AREA	
E0	FC1	TBC
E2	DPTR1	TBC
E6	FC2	TBC
E8	DPTR2	TBC
EC	FC3	TBC
EE	DPTR3	TBC
F2	FC4	TBC
F4	DPTR4	TBC
F8-FE	ZERO	Host

Figure 2-3. Initialzation Table (Sheet 2 of 2)

2.2.4 Initial Pad Timer Preset (PTP)

The initial pad timer preset (PTP) value is used to set the length and pattern of the preamble, and the minimum number of preamble octets transmitted between frames. This register must be set at initialization time by the host CPU according to the format shown below. The SET PTP command must be used after initialization to modify this register if desired. The type of physical layer used in the node determines the value to which the PTP is initialized. See **3.5.3.4 SET PAD TIMER PRESET REGISTER (PTP)** for more details on the PTP.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
78	m	m	m	m	m	b	b	b	p	p	p	p	p	p	p	p

Where:
m = Minimum number minus one of preamble octets transmitted between frames.
b = Minimum number of preamble octets minus two transmitted before frame after silence.
p = Pattern of preamble octet.

2.2.5 Initial In_Ring Desired

The initial in_ring desired parameter is a Boolean located in bit 15. A one indicates that the TBC should be a member of the token passing logical ring. After initialization the in_ring desired parameter may be modified using the SET/CLEAR IN_RING DESIRED command.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
7A	IR	AL	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2.2.6 Initial Address Length

The initial address length is a Boolean located in bit 14. A one indicates a 48-bit address and a zero indicates a 16-bit address. Note that the MAP specification specifies 48-bit addresses only.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
7A	IR	AL	0	0	0	0	0	0	0	0	0	0	0	0	0	0

2.2.7 Response Destination Address Pointer

The response destination address pointer points to a frame descriptor into which the TBC writes a received RWR frame's source address in the response frame's destination address field. See **4.4 REQUEST WITH RESPONSE (RWR) TRANSMISSION** for details on the RWR mechanism.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
84	MSB															
	Response Destination Address Pointer — High Word															
86																LSB
	Response Destination Address Pointer — Low Word															

2.2.8 Response Pointer

The response pointer contains a 32-bit address which points to the frame descriptor of the response to be sent when a request with response frame is received. The response pointer is used automatically only in predefined response mode. If the TBC is not in predefined response mode, the logical link control (LLC) must generate a response and update the response pointer. In that case,

the response pointer is not valid until the RESPONSE READY command is issued to the TBC by the host. See **4.4 REQUEST WITH RESPONSE (RWR) TRANSMISSION** for details on the RWR mechanism.

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
88	MSB							Response Pointer — High Word								
8A								Response Pointer — Low Word								LSB

2.2.9 Request with Response Pointer

The request with response (RWR) pointer contains a 32-bit address which points to the frame descriptor of the request with response frame that was received. See **4.4 REQUEST WITH RESPONSE (RWR) TRANSMISSION** for details on the RWR mechanism.

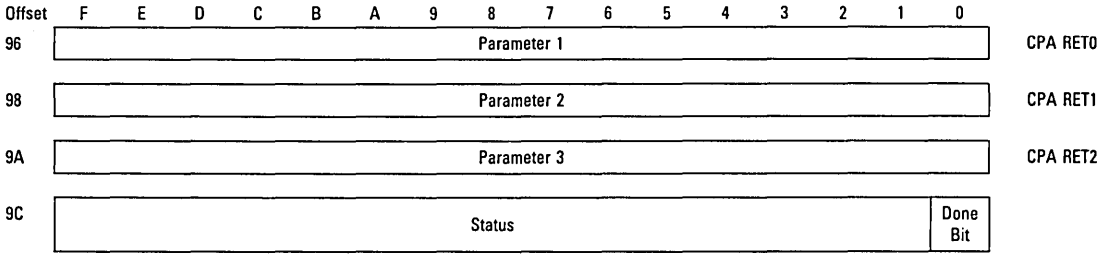
Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
8C	MSB							RWR Pointer — High Word								
8E								RWR Pointer — Low Word								LSB

2.2.10 Command Parameter Area

The command parameter area is located in words 90 through 9C of the initialization table. This area is used to read values from, and set values into, the TBC. The command parameter area format consists of the command area and the command result area. The most frequent usage of the command parameter area is by the SET ONE WORD, SET TWO WORDS, and READ VALUE commands. For these commands, the command area is a three word area with the first word containing the parameter opcode to be set or read. The other two words contain the new parameter when setting the value of that parameter. For a description of the parameter opcodes see Tables 3-7 and 3-8. The format of the command parameter area when using other commands is defined in the specific command description in **SECTION 3 COMMANDS.**

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
90	0	0	0	0	0	0	0	0	Opcode								CPA VAL0
92	Parameter 1 (If Needed)																CPA VAL1
94	Parameter 2 (If Needed)																CPA VAL2

The command result area is a four word area. The first three words contain the returned value of the parameter which was requested via a READ VALUE command and the fourth word contains the status. The done bit in the status word is set by the TBC when it is finished processing any command except RESET and LOAD INITIALIZATION TABLE FC. In order to use the done bit as an indicator of command completion, it must be cleared prior to issuing the TBC a command. The meaning of the status contained in offset 9C of the CPA is detailed under each command if used. The format of the command result area is:



2.2.11 Interrupt Status Words

Interrupt status bits are updated continuously by the TBC as status changes. To change a status bit, the TBC reads the appropriate word of the status area, updates the proper bit, and then writes the word back to the memory area. If the corresponding status bit is already set, then no action is taken by the TBC. The interrupt status word, combined with the interrupt status mask, determines whether an interrupt will be generated. If a special event occurs and the corresponding bit in the interrupt mask is zero, then an interrupt request will not be generated; however, the corresponding status bit will be set to one. If a special event occurs and the corresponding bit in the interrupt mask is one, then an interrupt will be generated, and the corresponding status bit will be set to one. To clear the interrupt status words, the CLEAR INTERRUPT STATUS command must be issued.

2.2.11.1 INTERRUPT STATUS WORD 0. The format of interrupt status word 0 is shown below:

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
A8	RXDF	RXRWR	TXDF	TXRDF	TXRWR	UNR	OVER	BDPL	FDPL	BAERR	TP	TSK	TXQE	BDPE	FDPE	TCC

TCC — TBC Command Complete
This bit is set upon completion of execution of all commands except RESET, LOAD INITIALIZATION TABLE FC, and CLEAR INTERRUPT STATUS.

FDPE — Frame Descriptor Pool Empty
This bit is set after the last frame descriptor (FD) in the free frame descriptor pool is used and linked to one of the receive queues.

BDPE — Buffer Descriptor Pool Empty
This bit is set when the TBC accesses the last buffer descriptor (BD) in the free buffer descriptor pool. The TBC does not use the last BD until the host has linked additional BDs to the pool. The TBC is able to receive only the header information (FC, DA, SA) from data frames when the BD pool is empty as long as free frame descriptors are available.

TXQE — Transmit Queue Empty
This bit is set after the TBC sets the confirm frame descriptor (CFD) bit of the last FD in any of the transmit queues. The queue must be reinitialized by the START command before transmitting again.

TSK — Token Skipped
This bit is set when the TBC hears a valid token frame being passed from a station "before" the TBC to a station "after" the TBC (i.e., SA>TS>DA or DA>SA>TS or TS>DA>SA). It is not affected by whether the TBC is in_ring. If the event occurs when the TBC is in_ring, an error has occurred.

TP — Token Passed

This bit is set when the TBC believes it has successfully passed the token to its successor station. When this event occurs, the TBC stores the token rotation time (TRT) value into the last token_rotation_time and begins a new TRT measurement.

BAERR — Bus/Address Error

This bit is set when a bus or address error occurs in a TBC DMA cycle. Bus error is indicated on the $\overline{BEC}$ pins, while an address error is generated when the TBC is bus master and $\overline{CS}$ or $\overline{IACK}$ is asserted, indicating that the TBC has attempted to address itself.

If a bus or address error occurs, the TBC stops transmitting and receiving data frames. If the TBC has the token at this time, the TBC will pass the token to its successor. If the TBC does not have a successor, then it will try to find a new successor. Next, the TBC will dump all four DMA pointers and their function codes into the dump area in the initialization table (offset E0 through F4) and execute the interrupt routine if the corresponding bit is set in the interrupt mask word. Note that the pointer which caused the bus/address error may have been incremented by one or two before its value was dumped. If a bus error occurs during reception, both free pools are disrupted and must be reinitialized via SET TWO WORDS commands as part of the bus error handling routine.

The host can cause the TBC to resume full operation by issuing a CLEAR INTERRUPT STATUS command to clear the bus/address error bit. This command should be given only after the steps described below have been followed:

- The host has dealt with the cause of the bus/address error.
- The host has given the TBC new pools of free FDs and free BDs.
- The host has enabled the TBC to resume transmitting by issuing the START or RESTART command if the TBC has more to transmit.

If a second bus/address error occurs before the host has dealt with the first one, (i.e., the appropriate CLEAR INTERRUPT STATUS command was not given), then the TBC enters an endless loop of severe interrupts and waits for RESET. If the TBC was in the OFFLINE state when the first bus/address error occurred, it will immediately enter the severe interrupt state.

FDPL — Frame Descriptor Pool Low

This bit is set when the TBC accesses an FD in the free FD pool whose W_FD (warning frame descriptor pool low) bit is set. The W_FD bit is intended to be a warning that the FD pool is running low and that more FDs should be added soon. The host decides in which FD to set the W_FD bit.

BDPL — BD Pool Low

This bit is set when the TBC accesses a BD in the free BD pool whose W_BD (warning buffer descriptor pool low) bit is set. The W_BD bit is intended to be a warning that the BD pool is running low and that more BDs should be added soon. The host decides in which BD to set the W_BD bit.

OVER — Overrun

This bit when set indicates that the TBC attempted to write to a full FIFO while receiving. This means that the TBC was not able to empty the FIFO fast enough. If this occurs frequently, it may indicate that the TBC is not receiving sufficient host bus bandwidth.

UNR — Underrun

If the TBC transmit machine attempts to read from an empty FIFO this bit will be set and an abort sequence will be sent by the TBC. This indicates that the TBC was not able to fill the FIFO fast enough. If this occurs frequently, it may indicate that the TBC is not receiving sufficient host bus bandwidth.

TXRWR — Transmitted Request With Response Frame

This bit is set when the TBC has finished processing a transmitted RWR frame and its response. This does not necessarily indicate that the frame was successfully transmitted or that an appropriate response frame was received. Setting this bit indicates that the status word of the FD of the RWR frame contains the detailed status of the frame.

TXRDF — Transmitted Response Data Frame

This bit is set when the TBC has finished processing a transmitted response data frame after receiving a RWR data frame. This does not necessarily indicate that the frame was successfully transmitted. Setting this bit indicates that the status word of the FD of the response frame contains the detailed status of the frame.

TXDF — Transmitted Data Frame

This bit is set when the TBC has finished processing a transmitted non-RWR frame or non-response data frame. This does not necessarily indicate that the frame was successfully transmitted. Setting this bit indicates that the status word of the FD of the frame contains the detailed status of the frame.

RXRWR — Received Request with Response Frame

This bit is set when the TBC accepts a valid non-retry RWR data frame, see **4.4 REQUEST WITH RESPONSE (RWR) TRANSMISSION** for the definition of retry. This bit is set only after the confirmation word has been written to the FD, the FD has been linked to the appropriate receive queue, and the TBC has written a pointer to the FD in the RWR pointer field of the initialization table. If a RWR frame addressed to the TBC is received with an error (e.g., noise, FCS error, overrun, lack of BDs), this bit will not be set even if the frame is accepted and linked to the appropriate receive queue. If such a frame is accepted, the RX data frame bit will be set although the frame control is that of a RWR frame.

RXDF — Received Data Frame

This bit is set when the TBC accepts a non-RWR data frame or an invalid request with response frame, after the confirmation word has been written to the FD and the FD has been linked to the appropriate receive queue.

2.2.11.2 INTERRUPT STATUS WORD 1. The format of interrupt status word 1 is shown below:

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
AC	MODER	*	LAS	TCE	BIEX	WAS	NRES	RECL	UNEXF10	SA	UEXF6	NS1	NS	SC	FTX	DMAD

*Reserved

DMAD — Duplicate MAC Address Detected

This bit is set when the TBC receives a control frame with SA equal to TS, when it is sure that this frame is not the echo of the TBC's own transmission. This means that there is another station in the ring with the same TS address as this station. Upon detecting such an event, the TBC sets this bit and goes OFFLINE.

FTX — Faulty Transmitter

This bit is set when the value of transmitter fault count stored in the private area exceeds seven. The value of the transmitter fault count is incremented each time the station sequences to the end of the token contention process and loses or fails to pass the token to any successor. Neither of these failures occurs in normal operation. The value of transmitter fault count is reset to zero if the station either enters the ring or successfully solicits a new successor, since such an event indicates that another station correctly heard a transmission from this station. Upon detecting the event of faulty transmitter, the TBC sets this bit and goes OFFLINE.

SC — Successor Changed

This bit is set when the TBC changes its successor. The new successor's address can be read via the READ VALUE command. This allows the network to maintain a "live list" of the stations in the logical ring.

NS — No Successor

NS1 — No Successor 1

When the TBC realizes that it has no successor, it sets either the no successor or the no successor 1 bit. The TBC can have no successor when it receives a set successor frame with data unit equal to TS, which means that this is the only station in the ring (no successor 1); or when it fails to pass the token to any successor, or when it gets the IDLE command and performs the transition from OFFLINE to IDLE (no successor). Note that no successor 1 represents normal operation while no successor, when not after the IDLE command, may indicate a problem in the network.

UEXF6 — Unexpected Frame 6

This bit is set when the TBC hears an unexpected frame when expecting a response to a transmitted request with response frame. An unexpected frame in this case is any valid frame which is not a response frame addressed to this station. Upon detecting this event, the TBC sets this bit and goes to IDLE without passing the token.

SA — Solicit Any Arc of the Access Control Machine (ACM) Performed

This bit is set when the TBC thinks it has a successor but reaches the solicit any phase of the pass token procedure. This means that the successor failed to respond to the token twice, and no station responded to two who_follows queries. The TBC sends a solicit any frame and sets this bit.

UNEXF10 — Unexpected Frame 10

The TBC sets this bit when it executes the "unexpected frame 10" transition in the IEEE 802.4 access control machine (ACM). It occurs when the TBC attempted to solicit a new successor and while waiting for a response, heard either a data frame not sent by itself, a set successor frame not addressed to it, or another type of control frame not sent by itself. When this event occurs, it indicates a protocol error, possibly a duplicate token situation. After setting this bit the TBC will go to IDLE without passing the token.

RECL — Receive Claim Token

This bit is set when the TBC hears a claim token frame sent by another station. This means that some other station thinks it is the only active station and wants to build a new logical ring.

NRES — No Response Received (ACM in the Await Response State)

This bit may be used as the MAC "heartbeat" indication to provide a periodic check on the functioning of the MAC sublayer. The heartbeat indication should occur periodically in a station where in_ring_desired remains true and sole_active_station is false, in other words a station which is a member of an active logical ring. If the heartbeat indication does not occur periodically, a monitor may assume that possibly the station's MAC is malfunctioning or some other catastropohic network failure has occurred. Also, if the bit is set too frequently, the maximum inter_solicit count may be incremented since no new station wants to enter the ring.

WAS — Win Address Sort

This bit is set when the TBC wins the claim token procedure and now is the one holding the token. The TBC sends data frames from access class 6 and then opens a response window to allow other stations to enter the ring.

BIEX — Bus Idle Timer Expired

This bit is set when the bus idle timer expires and the TBC has no frames to send, and does not want to be in_ring, or is the sole_active_station. It implies that nothing has been sent on the network for a long time, including tokens. This can happen if no station in the network wants to be in_ring, or the ring has collapsed, or if there is a fault in this station's receiver.

TCE — Threshold Counter Exceeded

This bit is set if any one of the statistic counters in the initialization table exceeded its threshold value. See **2.2.13 Statistics** for more detail.

LAS — Lose Address Sort

The TBC sets this bit when after six or seven slot times of silence, the TBC enters the claim token procedure, but does not win. The TBC returns to the IDLE state. This bit, along with the receive claim token, win address sort, and bus idle timer expired bits indicates that the logical ring, if there was one, has collapsed.

MODER — Modem Error

This bit is set if the physical layer has signalled a modem detected error. The TBC monitors error indication from the modem while in the idle state, or in the use token state. The modem error bit may also be set by the TBC if indicated from the physical layer during the receive or transmit test if run in external loopback. During transmission, this bit usually indicates that the modem detected that the TBC has transmitted for more than a half second. If an intelligent modem is used, then more information can be obtained via the physical command. Upon detecting this event, the TBC sets this bit and goes to OFFLINE.

2.2.12 Interrupt Masks

Interrupt masks have the same format as interrupt status words. If an interrupt status condition event occurs and the interrupt status bit is not already set, it causes an interrupt if the appropriate mask bit is set to one. An interrupt vector is then requested by the host processor and the host processor checks the interrupt status words to determine what caused the interrupt. If the mask bit is set to zero and the event occurs, no interrupt will be generated; however, the corresponding status bit in the interrupt status word will be set.

The format of interrupt mask 0 is shown below:

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
AA	RXDF	RXRWR	TXDF	TXRDF	TXRWR	UNR	OVER	BDPL	FDPL	BAERR	TP	TSK	TXQE	BDPE	FDPE	TCC

The format of interrupt mask 1 is shown below:

Offset	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
AE	MODER	*	LAS	TCE	BIEX	WAS	NRES	RECL	UNEXF10	SA	UEXF6	NS1	NS	SC	FTX	DMAD

*Reserved

2.2.13 Statistics

The initialization table contains statistics about the operation of the network. The statistic counters are 16-bit wrap-around counters. Every counter has a threshold variable. Whenever a counter is incremented, the TBC checks whether its value has exceeded the threshold value set by the host.

If the threshold value has been exceeded, the status bit threshold counter exceeded (TCE) in interrupt status word 1 is set. If enabled by the TCE bit in interrupt mask word 1, an interrupt is also generated. A threshold value of zero disables the interrupt mechanism, however the counters are still incremented. The threshold mechanism allows software to map the 16-bit counters into larger counters (32 bits), or to be notified if a large number of errors occurs. Counters are not automatically cleared when the threshold value is reached. The collection of two of these statistics which are number of tokens passed and number of tokens heard may be disabled using the SET MODE 1 command (see **3.3.1 SET MODE 1 Command**).These counters are described in the following paragraphs.

2.2.13.1 NUMBER OF TOKENS PASSED. This statistic is the number of tokens that have been passed by this station. The counter is incremented when the TBC thinks it successfully passed the token to its successor in the ring. In normal operation, this is equal to the number of tokens passed to the TBC. This statistic enables software to calculate the average token rotation time. The collection of this statistic may be disabled using the SET MODE 1 command.

2.2.13.2 NUMBER OF TOKENS HEARD. This statistic represents the number of tokens this station has heard on the network with DA not equal to TS and SA not equal to TS, that is the number of tokens that have gone by. This counter, along with the tokens passed counter, the token skipped and token passed status bits in interrupt status word 0, may be used to estimate the number of stations in the logical ring. The collection of this statistic may be disabled using the SET MODE 1 command.

2.2.13.3 NUMBER OF PASSES THROUGH NO SUCCESSOR 8. This statistic indicates the number of times this station has gone through the no successor 8 arc in the state machine. This happens when the TBC fails to pass the token and does not succeed in finding a new successor station. The counter is incremented only if the TBC thinks it is not the only active station in the network. A significantly large value in this counter may indicate a "faulty" transmitter condition in this station.

2.2.13.4 NUMBER OF WHO FOLLOWS. This statistic is the number of times this station has had to look for a new next station to pass the token to. This frame is sent as part of the TBC's effort to pass the token to its former successor's successor if the original successor station does not respond to the token. This counter is incremented by two every time a failure occurs.

2.2.13.5 NUMBER OF TOKEN PASS FAILED. This statistic indicates the number of token pass failed transitions when pass state is equal to pass token. Upon failing to pass the token, the TBC tries to send a second token (pass state equals repeat pass token). If this effort fails too, this counter is not incremented again; but the TBC will then send a who follows frame and the who follows query counter will be incremented.

2.2.13.6 NUMBER OF FRAMES TOO LONG (>8K BYTES). This statistic is the number of received frames that are too long (>8K bytes, an IEEE 802.4 parameter). If frames are longer than 8K, they may be accepted if the appropriate bit is set in the receive frame status error mask (see **4.1.1.2 RECEIVE STATUS WORD** for a description) in the initialization table.

2.2.13.7 NUMBER OF NO FD/BD ERRORS. This statistic counts the number of frames that were not received because there were not enough frame descriptors or there were not enough buffers.

When there are not enough buffers, frames may be accepted if the appropriate bit is set in the receive frame status error mask (see **4.1.1.2 RECEIVE STATUS WORD** for a description) in the initialization table.

2.2.13.8 NUMBER OF OVERRUNS. This statistic represents the number of times the TBC detected a FIFO overrun during receive. Frames may be stored into memory up to the overrun error if the appropriate bit is set in the receive frame status error mask (see **4.1.1.2 RECEIVE STATUS WORD** for a description) in the initialization table.

2.2.14 Modem Error Counters

The following three counters are the number of noise bursts detected that occurred when noise was not expected. Noise may be expected in some procedures in the protocol due to collisions. These counters do not track such noise bursts, but only noise bursts that are due to errors or unexpected noise on the medium.

2.2.14.1 NON-SILENCE. Non-silence is the number of received periods of non-silence.

2.2.14.2 FCS ERRORS. FCS errors track the number of received frames with FCS (or CRC) errors and the E-bit reset.

2.2.14.3 E-BIT ERRORS. E-bit errors count the number of received frames with the E bit set in the end delimiter. The E bit, or error bit, is set by the regenerative repeater (headend remodulator), when the headend detects a FCS error on the forward channel. If this error occurs, frames may be accepted if the appropriate bit is set in the receive frame status error mask (see **4.1.1.2 RECEIVE STATUS WORD** for a description) in the initialization table.

2.2.14.4 FRAME FRAGMENTS. This counter represents the number of frame fragments (start delimiter (SD) not followed by a valid end delimiter (ED)). A valid frame consists of only data (zero or one MAC symbols) between the SD and the ED. If an SD is detected and then, before a valid ED, the TBC detects either silence, non data (not part of the aligned ED), or bad signal, then this counter is incremented. Note that this includes abort sequences.

2.2.15 DMA Dump Area

The DMA dump area contains the pointers and function codes of the four DMA channels when a bus/address error occurs (see **2.2.11 Interrupt Status Words**). The dump area can be used to check all of the pointers and to see if one of them is out of sequence. This should be the pointer that caused the bus/address error.

SECTION 3
COMMANDS

The host processor controls the TBC and the modem through the command mechanism. This command mechanism is composed of the command register (CR), the semaphore register (SR), and the command parameter area (CPA) in the initialization table. The 8-bit, write only, command register is used to write commands to the TBC. The 8-bit, read only, semaphore register is used to notify the host that execution of either the RESET or LOAD INITIALIZATION TABLE FUNCTION CODE commands have been completed. The actual completion of a command except for RESET, and LOAD INITIALIZATION TABLE FUNCTION CODE is indicated by the TBC setting of the 'done bit' in the command parameter area in the initialization table. Another indication of command completion is the TBC command complete bit (TCC) in the interrupt status word 0 which, if enabled, can generate an interrupt to the host. The TBC command set is divided into the following categories:

Initialization,
Set Operation Mode,
TX Data Frames,
Set/Read Value,
Test,
Notify TBC, and
Modem Control.

Table 3-1 shows the commands for each category, while Table 3-2 illustrates each command's encodings.

Table 3-1. TBC Commands by Categories

INITIALIZATION LOAD INITIAL TABLE FUNCTION CODE INITIALIZE OFFLINE IDLE RESET	**TEST** INTERNAL/EXTERNAL LOOPBACK MODE RECEIVER TEST TRANSMITTER TEST HOST INTERFACE TEST FULL DUPLEX LOOPBACK TEST
SET OPERATION MODE SET MODE 1 SET MODE 2 SET MODE 3 SET/CLEAR IN_RING DESIRED	**NOTIFY TBC** CLEAR INTERRUPT STATUS RESPONSE READY
TX DATA FRAMES STOP RESTART START	**MODEM CONTROL** PHYSICAL END PHYSICAL
SET/READ VALUE READ VALUE SET FUNCTION CODE OF BUFFER DESCRIPTORS SET FUNCTION CODE OF FRAME DESCRIPTORS SET FUNCTION CODE OF RX AND TX DATA BUFFERS SET PAD TIMER PRESET (PTP) REGISTER SET ONE WORD SET TWO WORDS	

Table 3-2. TBC Commands

Command	7	6	5	4	3	2	1	0	Hex
ILLEGAL	0	0	0	X	X	X	X	X	00-1F
SET MODE 1	0	0	1	0	Y	Y	Y	Y	20-2F
SET MODE 2	0	1	0	Y	Y	Y	Y	Y	40-5F
SET MODE 3	0	1	1	Y	Y	Y	Y	Y	60-7F
SET FC BD	1	0	0	0	0	0	0	1	81
SET FC FD	1	0	0	0	0	0	1	0	82
SET PAD TIMER PRESET (PTP)	1	0	0	0	0	0	1	1	83
START	1	0	0	0	0	1	0	0	84
SET TWO WORDS	1	0	0	0	0	1	0	1	85
SET ONE WORD	1	0	0	0	0	1	1	0	86
SET FC DATA	1	0	0	0	0	1	1	1	87
CLEAR INTERRUPT STATUS	1	0	0	0	1	0	0	0	88
READ VALUE	1	0	0	1	0	0	0	0	90
PHYSICAL	1	0	1	0	0	Y	Y	Y	A1-A7
END PHYSICAL	1	0	1	0	1	0	0	0	A8
RECEIVE TEST	1	0	1	1	0	0	0	0	B0
TRANSMIT TEST	1	0	1	1	0	1	Y	0	B4, B6
FULL-DUPLEX LOOPBACK TEST	1	0	1	1	1	0	0	0	B8
HOST INTERFACE TEST	1	0	1	1	1	1	0	0	BC
INT/EXT LOOPBACK MODE	1	1	0	0	Y	1	0	0	C4, CC
RESPONSE READY	1	1	0	0	0	1	0	1	C5
RESTART	1	1	0	0	0	1	1	0	C6
INITIALIZE	1	1	0	0	0	1	1	1	C7
SET/CLEAR IN_RING DESIRED	1	1	0	0	1	Y	1	1	CB, CF
LOAD INIT TABLE FC	1	1	0	1	0	0	0	Y	D0, D1
STOP	1	1	1	0	0	0	0	0	E0
IDLE	1	1	1	1	0	0	0	0	F0
OFFLINE	1	1	1	1	1	0	0	0	F8
RESET	1	1	1	1	1	1	1	1	FF

CODING OF COMMANDS
X — Don't Care
Y — Information Bit

Commands are executed by the TBC either while uninitialized, in the OFFLINE or IDLE modes, or in the use_token state in between transmitting and receiving frames. Commands which have an opcode not specified in Table 3-2 are reserved. In most cases, if one of these reserved opcodes is loaded into the command register the TBC will execute the command with the legal opcode closest to the reserved one. However, this behavior cannot be guaranteed and care should be taken to always load the command register with legal opcodes.

3.1 REGISTERS

The following registers are writable by the host CPU: Command Register (CR), Data Register (DR), and Interrupt Vector Register (IV). Another register, the semaphore register (SR), is only readable

by the host. Table 6-1 shows how the TBC registers are accessed during a write cycle for an 8-bit data bus and Table 6-2 shows how the TBC registers are accessed using a 16-bit data bus (these tables are located in **SECTION 6 BUS OPERATION**).

3.1.1 TBC Register Map

The register map for the TBC is shown in Table 3-3. Byte one (least significant byte) of the first word holds the 8-bit command register which is also used as the semaphore register. Byte one of the second word holds the interrupt vector register. The 32-bit data register is in the next four bytes, DR0 being the least significant byte and DR3 being the most significant byte of that register.

Table 3-3. Register Map

	D15 D8	D7 D0
BASE		CR/SR
BASE +2		IV
BASE +4	DR3	DR2
BASE +6	DR1	DR0

3.1.2 Command Register (CR)

The 8-bit command register is used by the host processor to send commands to the TBC. There are 28 valid commands in seven categories that can be issued by the host processor to the TBC.

3.1.3 Data Register (DR)

The 32-bit data register is used as a data input port to receive the initialization table pointer and function code. The TBC also uses the data register during DMA transfers. See **6.2 DMA OPERATION** for details on DMA transfers. The format of the 32-bit data register is shown in Table 3-4.

Table 3-4. Data Register Format

31 24	23 16	15 8	7 0
DR3	DR2	DR1	DR0

3.1.4 Interrupt Vector Register (IV)

The 8-bit interrupt vector register is used to store the 8-bit interrupt vector returned to the host by the TBC for the interrupt acknowledge (IACK) cycle. This interrupt vector contains the address of the software routine to be executed by the host when an interrupt occurs. The TBC uses two interrupt status words located in the initialization table to inform the host about events which occurred. The TBC's actions after such an interrupting condition occurs are described in **2.2.11 Interrupt Status Words**. The CLEAR INTERRUPT STATUS command which is used by the host to clear interrupts in the interrupt status words is described in **3.7.1 CLEAR INTERRUPT STATUS Command.** In order to use the interrupt mechanism provided by the TBC, the host must load the upper six bits of the interrupt vector into the IV at initialization. The lowest two bits are a prioritized code that indicates which of two categories requested the interrupt. These two bits are loaded by the TBC and are '01' to indicate a bus/address error which occurred twice in a row (i.e., a

severe interrupt), that is the host did not take appropriate action after the first bus/address error indication. See **2.2.11 Interrupt Status Words** for more details. For all other interrupts, the two least significant bits of the interrupt vector register are set to '00'. The interrupt vector register is set to '0F' after reset and remains '0F' until written to for the first time. Table 3-5 shows the format of the interrupt vector.

Table 3-5. Interrupt Vector

7	6	5	4	3	2	1	0
Furnished by Host from IV						Loaded by TBC	

3.1.5 Semaphore Register

This register is written to by the TBC and is read by the host processor. When any command is written to the TBC command register, the TBC indicates that it has accepted the command by setting the semaphore register to 'FE'. In order to determine when the TBC has completed a command, the done bit in the status word of the CPA located in the initialization table may be checked, if it was previously cleared, for all commands except RESET and LOAD INITIALIZATION TABLE FUNCTION CODE. The completion of these two commands is indicated by the semaphore register changing to 'FF'. The semaphore register has the same address as the command register.

3.2 INITIALIZATION

Commands in the initialization category are used to initialize the TBC and to change the access control machine (ACM) state from OFFLINE to IDLE. The following routine is used to initialize the TBC by the host processor. For every command except the RESET and LOAD INITIALIZATION TABLE FUNCTION CODE commands, the done bit in the status word of the command parameter area may be checked to ensure the instruction was executed by the TBC if the host processor has cleared the fourth word of the command result area before issuing the command.

Prepare the Initialization Table
Issue RESET[1] Command
Wait until Semaphore Register is 'FF'
Initialize the Interrupt Vector Register
Load the Function Code of the Initialization Table Pointer into DR0[2]
Issue LOAD INITIALIZATION TABLE FC[3] command
Wait until Semaphore Register is 'FF'
Load the Initialization Table Pointer into DR
Issue INITIALIZE Command
Issue SET MODE 1 Command
Issue SET MODE 2 Command
Issue SET MODE 3 Command
Initialize Modem
Issue SET FC FOR THE BUFFER DESCRIPTORS Command[2]
Issue SET FC FOR THE FRAME DESCRIPTORS Command[2]
Issue SET FC FOR RX and TX DATA BUFFERS Command[2]
Issue IDLE Command

NOTES:
1. After reset, the TBC is in default mode where all bits in the SET MODE commands are cleared, except for the halt generator enable mode which is set (see **3.3.3 SET MODE 3 Command**).
2. Necessary only if using function codes.
3. This command also chooses the bus width of 8 or 16 bits. Note that prior to this command the bus width was 8 bits.

3.2.1 LOAD INITIALIZATION TABLE FUNCTION CODE Command

The LOAD INITIALIZATION TABLE FUNCTION CODE command is the first command in the initialization sequence of the TBC. This command loads the 4-bit initialization table function code through the data register (DR0) and sets the bus width to 8 or 16 bits. If function codes are not used, no value needs to be loaded into DR0, however if the bus width is 16 bits this command still needs to be issued. If the desired bus width is eight bits and no function codes are used, this command is unnecessary. The only confirmation this command returns is an 'FF' in the semaphore register. The coding for this command is D0 or D1 and its format is shown below.

7	6	5	4	3	2	1	0
1	1	0	1	0	0	0	BUSW

Where:
BUSW = Bus Width
0 = 8-Bit Bus
1 = 16-Bit Bus

3.2.2 INITIALIZE Command

INITIALIZE is the second command in the initialization sequence of the TBC. This command must only be given while in the OFFLINE state after a RESET. Before giving this command, the host must load the initialization table in shared memory and write the pointer for the initialization table into the DR register.

The INITIALIZE command causes the TBC to load the initialization table pointer from the DR register, load the private area function code and pointer from the initialization table, load the operational TBC parameters from the initialization table, and initialize the private area. The TBC also performs a self test of its internal resources upon execution of this command. The TBC will return confirmation of this command in the CPA (offset 9C of the initialization table), provided this word was cleared prior to command execution. If no errors are detected during the self test, a value of one will be returned in the status word of the CPA. Otherwise, a value of 0011 hex will be stored in the status word of the CPA. If a bus/address error occurs during execution of this command, a severe interrupt will be generated requiring a RESET of the TBC. The coding for this command is C7 and its format is shown below:

7	6	5	4	3	2	1	0
1	1	0	0	0	1	1	1

3.2.3 OFFLINE Command

The OFFLINE command causes the TBC to leave its present state and enter the OFFLINE state. This command is typically used to end receive and transmit tests (see **3.6.3 FULL-DUPLEX LOOPBACK TEST Command** and **3.6.4 TRANSMITTER TEST Command**) and when the TBC is not in_ring. Execution of this command clears the any_send_pending flag (IEEE 802.4 boolean) and resets the receive and transmit machines.

If the TBC is transmitting when the OFFLINE command is issued, the current frame may not be completed; that is, the end delimiter may not be sent. If the TBC is in_ring when this command is given, the TBC will immediately drop out of the ring without notifying other stations or passing the token. The OFFLINE command should only be issued when the TBC is in_ring if, for some reason, the proper exit method is not successful. The proper way to exit the ring is to issue the STOP command, clear in_ring desired, and wait for a no_successor status from the TBC. Then

the OFFLINE command may be given safely. The coding for this command is F8 and its format is shown below:

7	6	5	4	3	2	1	0
1	1	1	1	1	0	0	0

WARNING

The free FD and BD pool pointers (see **SECTION 4 BUFFER STRUCTURES)** may be incorrect after issuing the OFFLINE command if any data frame was received since they were last initialized. They must be reinitialized (SET TWO WORDS command) before giving the IDLE or RECEIVE TEST command.

3.2.4 IDLE Command

The IDLE command causes the TBC to go from OFFLINE to the IDLE state. The IDLE command must only be given when the TBC is in the OFFLINE state (i.e., not in_ring). The free frame descriptor and free buffer descriptor pools should be initialized before giving this command. The IDLE command is used, for example, after initialization, testing, or physical management.

The IDLE command causes the TBC to:

Clear any_send_pending (802.4 boolean),

Reset the receive and transmit machines,

Enable the receive machine,

Set noise expected (see **2.2.13 STATISTICS**),

Zero the last_token_rotation_timer (TRT) variable located in the private area,

Zero the internal token_rotation_timer (TRT) counter,

Set the no_successor status bit,

Zero the transmitter fault count located in the private area,

Load free FD and BD pool pointers from the private area to the TBC,

Return command confirmation, and

Enter the IDLE state.

The coding for this command is 'F0' and its format is shown below:

7	6	5	4	3	2	1	0
1	1	1	1	0	0	0	0

3.2.5 RESET Command

This command is equivalent to a hardware reset and is used to reset the TBC. RESET may be issued to the TBC with the semaphore register equal to 'FE' or 'FF'. It:

Sets the semaphore register to 'FE' hex,

Returns all modes to their default values of zero except for HLEN which defaults to one (see **3.3.3 SET MODE 3 Command**),

Sets the bus width to its default value (eight bits),

Resets the receive and transmit machines,

Sends silence symbols to the modem,

Zeros the internal TRT counter,

Zeros the internal copy of interrupt status word 0, and

Causes the TBC to believe the free FD pool is not empty and resets in_ring and NS known (802.4 booleans).

After completing the above, the TBC sets the semaphore byte to 'FF' hex when ready to receive the LOAD INIT TABLE FC command. There is no other indication that the command has been executed.

The coding for this command is 'FF' and its format is shown below:

7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1

3.3 SET OPERATION MODE

The four commands in this category are used to set the various operation modes and options in the TBC. The default setting is zero, or disabled, for all modes except for HLEN whose default is one (see **3.3.3 SET MODE 3 Command**). These commands are normally given only during initialization or testing.

3.3.1 SET MODE 1 Command

The default setting is zero for all modes described in this subsection. This command is normally given only as part of an initialization sequence or during testing.

7	6	5	4	3	2	1	0
0	0	1	0	CUFC	CACF	LSTT	PDRF

CUFC — Recognize Frames with Undefined Frame Control (not part of IEEE 802.4)

1 Recognize frames with undefined frame control as valid data frames

0 Do not recognize frames with undefined frame control as valid data frames

When disabled, frames with undefined frame control fields are treated as noise bursts or invalid frames and are not stored in memory. When enabled, the frames are treated as valid data frames and the TBC can support additional frame control fields to be defined by the user or by IEEE 802.4 in the future. This mode can also be useful in detecting error conditions on the network.

CACF — Copy all Control Frames

1 Copy all control and data frames to queue six

0 Copy only data frames to the appropriate queue (normal operation)

In the CACF mode, all control and data frames are copied to the data buffers if the necessary criteria for accepting a frame as in normal operation are met (destination address match, etc.). This mode is useful for monitoring network traffic. In this mode, all received frames

are treated as data frames of priority six even if their priority is lower. In this mode, the TBC must not be part of the logical ring since control frames are not recognized as control frames. For the TBC to be in promiscuous mode, this mode bit must be set and in addition, the individual address mask located in the private area should be set to zero while the group address mask should be set to all F's (see **APPENDIX B FRAME FORMATS AND ADDRESSING** for more details).

LSTT — Limited Statistics Tracking

1 Keep track of infrequent statistics only

0 Update all statistics

In limited statistics counter mode, the TBC updates all statistics except the token pass counter and the token heard counter located in the initialization table. The statistics kept in the private area are not affected by this mode (last token rotation time, elapsed time to access class 0, 2, 4, or 6).

PDRF — Predefined Response Mode

1 In predefined response mode

0 Not in predefined response mode

While in predefined response mode, when the TBC receives a valid request with response (RWR) frame, it immediateiy fetches and transmits a prepared response frame pointed to by the response pointer in the initialization table (offset 88). When not in this mode, the TBC notifies the host processor that a RWR frame has been received and waits for the host processor to prepare a response and issue a RESPONSE READY command. See **4.4 REQUEST WITH RESPONSE (RWR) TRANSMISSION** and **4.5 RECEPTION OF RWR FRAMES AND TRANSMISSION OF RESPONSE** for a complete description of RWR frames.

3.3.2 SET MODE 2 Command

The default setting is zero for all modes described in this subsection. This command is normally given only as part of an initialization sequence or during testing.

7	6	5	4	3	2	1	0
0	1	0	LBRM	IRND	SRLB	BRG	RSR

LBRM — Lower Bridge Mode

1 Recognize lower bridge mode

0 Do not recognize lower bridge mode

In lower bridge mode, the TBC accepts frames if the destination address ANDed with the individual address mask is not equal to this station's address ANDed with the individual address mask (DA AND IA mask not equal to TS AND IA mask). When this mode is disabled, the TBC accepts the frame only if equality is found between the two expressions. Similarly, when LBRM is enabled, group addresses are accepted if the DA does not match the group address (GA) mask. Reception of broadcast frames is not affected by this mode. Lower bridge mode is used in a hierarchical bridge architecture. Note that the lower bridge mode and the recognize source routing mode (bit 0) are mutually exclusive. If both modes are set, recognize source routing mode takes precedence. For more details on bridges, see **APPENDIX C BRIDGING.**

IRND — In_Ring Desired
1 In_ring desired
0 In_ring not desired

This bit determines the TBC's steady state condition when it is not OFFLINE and has no queued transmission requests. The initial value of this mode is set in the initialization table (offset 7A). This command should only be used at initialization time or for testing purposes. The preferred way to modify the in_ring desired flag is by using the SET/CLEAR IN_RING DESIRED command (see **3.3.4 SET IN_RING DESIRED Command**).

SRLB — Source Routing Limited Broadcast
1 In source routing limited broadcast mode
0 Not in source routing limited broadcast mode

This bit has meaning only when recognize source routing (RSR) is enabled. Source routing frames are of two types: broadcast and non-broadcast. A broadcast frame may be limited broadcast or unlimited broadcast. Unlimited broadcast frames are accepted by all bridges while limited broadcast frames are accepted only by bridges which are in limited broadcast mode. This option can be used to reduce the amount of traffic generated by broadcast frames or to specify default routing paths. More details may be found in **APPENDIX C BRIDGING.**

BRG — Bridge Delay Mode
1 Enable bridge delay mode
0 Disable bridge delay mode

This mode is used when the TBC transmits 'aliased' data frames; i.e., SA not equal to TS. This occurs in bridges, which can transmit frames originating on other segments. This bit should only be set in broadband networks where the TBC is located far enough from the head-end remodulator so that the entire echo of a minimal length frame transmitted by the TBC could be heard by it after it has completed transmission. This mode solves a problem caused by hearing a data frame with SA not equal to TS, which could cause the TBC to incorrectly believe that another station is transmitting. In this mode, the TBC will wait approximately one slot time before transmitting a solicit successor or token frame after transmitting a data frame, or before transmitting a non-retry request with response data frame.

RSR — Recognize Source Routing
1 Recognize source routing
0 Do not recognize source routing

This mode allows the TBC to act as a part of a source routing MAC bridge between interconnected networks. The meaning of the least significant bit of the source address of a frame having a value of one is not defined in the IEEE 802 standards. The source routing method uses this bit in data frames to indicate the presence of a routing information field in the frame which describes the segment routing of the frame. This is a non-address based routing scheme, meaning that the destination address does not itself contain information on where the destination station is located. When the least significant bit of the source address bit is zero, the frame is treated as an ordinary frame. If this bit is a one in a control frame, the frame is considered invalid and is treated as a noise burst. If this bit is a one in a data frame and if RSR mode is disabled, the TBC treats the frame as a normal frame (i.e., if the destination address matches the criteria set by the host, the frame is accepted and the routing information field is treated as normal data). If the RSR mode is enabled, the frame is accepted if the destination address matches the specified criteria or if the routing information field specifies that the frame is to be routed to another segment via the TBC. The routing information field is copied to the first data buffer like normal data. The length of the buffers in the free pool

should be at least 64 bytes in this mode. Note that the lower bridge mode (bit 4) and the recognize source routing mode are mutually exclusive. If both modes are set, recognize source routing mode takes precedence. See **APPENDIX C BRIDGING** for details.

3.3.3 SET MODE 3 Command

The default setting is zero for all modes described in this subsection except for the halt generator enable mode (HLEN) whose default is one. This command is normally given only as part of an initialization sequence or during testing.

7	6	5	4	3	2	1	0
0	1	1	RCDS	TCDS	HLEN	SWAP	PS3

RCDS — RX CRC to Data Storage Mode
- 1 Copy RX CRC as part of the data unit
- 0 Do not copy RX CRC as part of the data unit.

This mode causes the TBC to copy the 4-byte CRC of data frames to memory as part of the data unit. The CRC is counted in the reported data unit length. Regardless of this mode, the CRC is checked for correctness and, if erroneous, the TBC will not accept the frame unless so specified in the RX frame status error mask in the private area (offset 26).

TCDS — TX CRC Disable Mode
- 1 CRC not generated by the TBC for data frames
- 0 CRC generated by the TBC

This mode disables CRC generation on data frames transmitted by the TBC. In this case, the user must supply a correct CRC at the end of the data unit. According to the 802.4 standard, all frames are transmitted with the standard 32-bit frame check sequence (FCS) and the FCS or CRCis checked for correctness in all received frames. Received frames with CRC errors are treated as noise bursts by the access control machine (ACM). This mode applies to all data frames, i.e., there is not a separate indication for each frame. The TBC always generates and attaches the CRC for all control frames, even when this mode is on. This mode can be used for testing or for use with a non-standard, user specified CRC.

NOTE

In the latter case, receiving stations may identify data frames as noise bursts. The CRC error bit in the RX status error mask located in the private area must be set in order to accept such frames.

HLEN — Halt Generator Enable Mode
- 1 Halt generator enabled
- 0 Halt generator not enabled

This mode controls the maximum number of DMA transfers that the TBC performs in one DMA burst. If the user wishes to give the maximum memory bandwidth to the TBC, HLEN should be set to zero. This will allow the TBC to, for example, empty or fill the FIFO in one burst. If the user wishes to limit TBC DMA bursts, HLEN should be set to one. This will cause the TBC to release the bus after a maximum of eight memory cycles. In this case, if the FIFO is not empty or full, the TBC will request the bus again. After RESET, HLEN is enabled.

SWAP — Data Byte Swap Mode
- 1 Data in memory buffers organized with high order byte in higher address (Digital and Intel convention)

0 Data in memory buffers organized with high order byte in lower address (Motorola and IBM convention)

SWAP = 1 — Intel Convention

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
ODD (HIGHER) ADDRESS								EVEN (LOWER) ADDRESS							

SWAP = 0 — Motorola Convention

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
EVEN (LOWER) ADDRESS								ODD (HIGHER) ADDRESS							

In an 8-bit mode, this bit has no significance. Note that while the data organization of data buffers and the frame header in the FD may be selected by this option, all other parameters, pointers, and tables are organized according to the Motorola/IBM convention.

PS3 — Prescaler 3 Bit Mode

1 Prescaler of three bits for octet timers

0 Prescaler of six bits for octet timers

The following octet variables, which are either parameters or statistics, are located in the private area or in the initialization area of the initialization table and are subject to a prescaling of three or six bits as set by the PS3 bit:

Hi_priority_token_hold time	Parameter
Last token_rotation_time	Statistic
Token rotation time at start of access classes (0, 2, and 4)	Statistic
Target token rotation time for access classes (0, 2, and 4)	Parameter
Token rotation time at start of ring maintenance	Statistic
Target token rotation time for ring maintenance	Parameter
Ring maintenance timer initial value	Parameter

The prescaling mechanism provides a resolution of 8 or 64 octets for all the values listed above. The amount of prescaling must be chosen such that after prescaling, the parameters will fit into 15 bits. Thus, if the largest target rotation time parameter or the expected token rotation time is larger than $(2^{18})-1$ octet times, the prescaler size must be six bits. When rotation times are this large, the eight octet resolution is finer than needed, and the 64 octet resolution is sufficient. A value of FFFF for any one of the statistics listed above indicates an overflow.

3.3.4 SET/CLEAR IN_RING DESIRED Command

This command is used to change the value of in_ring desired during normal operation. This method of clearing, or setting, in_ring desired should be used rather than the SET MODE 2 command during normal operation. A description of the in_ring desired variable, which is an IEEE 802.4 boolean, is found in **3.3.2 SET MODE 2 Command**. The coding for this command is CB or CF and its format is:

7	6	5	4	3	2	1	0
1	1	0	0	1	SET/ CLEAR	1	1

Where:

1 Set In_Ring Desired

0 Clear In_Ring Desired

3.4 TX DATA FRAMES

Commands in this category are used to stop/restart transmission of data frames or to start transmission from an empty queue.

3.4.1 STOP Command

The STOP command causes the TBC to suspend transmission of data frames in all queues. This command does not affect the status of the four transmit queues (see **2.1.15 TX Queue Access Class Status**). If the TBC is holding the token when the STOP command is received, the TBC will complete transmission of the current frame and then pass the token. If in_ring desired is true, the TBC will remain in_ring or, if not currently in_ring, will continue to try to enter the ring. The STOP command is equivalent to clearing the any_send_pending boolean defined in the IEEE 802.4 standard. The coding for the STOP command is E0 and its format is shown below.

7	6	5	4	3	2	1	0
1	1	1	0	0	0	0	0

3.4.2 RESTART Command

This command restarts transmission of data frames by the TBC. RESTART sets the any_send_pending boolean and differs from the START command in that it only restarts transmission of data frames and does not affect the status of the four transmit queues. The RESTART command is used after transmission of data frames was suspended by an earlier STOP command. If the TBC is not in_ring when this command is given, the TBC will enter the logical ring if outstanding data frames remain to be transmitted. The coding for this command is C6 and its format is shown below.

7	6	5	4	3	2	1	0
1	1	0	0	0	1	1	0

3.4.3 START Command

This command is used by the host processor to start transmission only when a new frame has been added to an empty transmit queue. Before loading this command into the command register (CR), the host must initialize the command parameter area located in the initialization table. The first word of the CPA, CPA VAL0 (offset 90), must contain the code of the appropriate transmit queue. These are:

Queue	CPA VAL0 (Hex)
6	0028
4	0030
2	0038
0	0040

The second and third words of the CPA, CPA VAL1 and CPA VAL2 respectively, must contain a pointer to the new head of the transmit queue. This pointer is the address of the first frame descriptor in the queue, with the most significant word of the pointer in CPA VAL1.

The TBC, when issued this command, sets its internal any_send_pending status to true. It also zeroes the appropriate transmit queue status word in the private area. In other words, it sets the

status of the queue to enabled and not empty and writes the pointer to the appropriate head of transmit queue pointer entry in the private area.

This command should not be used to enable a disabled low priority (4, 2, 0) queue. The token rotation timers of disabled priority queues are not maintained. Therefore, a lower priority queue must be enabled at least one token rotation before queueing a frame for transmission on that queue. Lower priority queues are preferably enabled in the OFFLINE state by the SET ONE WORD command (see **3.5.3 SET VALUE Command**). Note that adding a frame to a transmit queue which is not empty and is enabled is an implicit request for transmission, therefore the START command is not needed in this case.

The coding of this command is 84 and its format is shown below.

7	6	5	4	3	2	1	0
1	0	0	0	0	1	0	0

3.5 SET/READ VALUE

Commands in this category are used by the host processor to read or set TBC parameters residing in the private area or on chip. These commands use the command parameter area in the initialization table to transfer parameters between the host and the TBC.

3.5.1 Command Parameter Area

The command parameter area (CPA) is a seven word memory area within the initialization table (offset 90 to 9C) used to set and read internal variables of the TBC which reside in the private area or on chip via the SET/READ VALUE commands. The CPA consists of the command area and the command result area.

The format of the command parameter area is shown in Figure 3-1. The command parameter area is a three word area located at offset 90 through 94 of the initialization table. The first word contains the opcode for the parameter to be set or read, see Tables 3-6 and 3-7 for the opcodes list, while the other two words contain the parameters to be passed from the host processor to the TBC.

```
INITIALIZATION TABLE    +90   CPA VAL0 (CODE)
                        +92   CPA VAL1
                        +94   CPA VAL2
                        +96   CPA RET0
                        +98   CPA RET1
                        +9A   CPA RET2
                        +9C   CPA RESULT (STATUS WORD)

CODE    — DEFINES THE CODE OF THE OPERATION IN READ/SET VALUE. THIS CODE IS ALSO USED FOR
          THE START COMMAND (SEE 3.4.3 START).
VAL     — DEFINES THE VALUE OF THE PARAMETER TO BE WRITTEN
RET     — CONTAINS THE RETURNED PARAMETER FROM THE TBC
RESULT  — CONTAINS THE STATUS
```

Figure 3-1. Command Parameter Area

Table 3-6. Read Value Opcodes

Code (Hex)	Length in Bytes	Name	Description	
00	2, 6	NS	Next Station's Address	802.4
04	2, 6	PS	Previous Station's Address	802.4
08	2	HI_PRI_THT	High_Priority_Token_Hold_Time	802.4
0A	2	LAST_TRT	Last Token_Rotation_Time	802.4
0C	2	START(4)	TRT at Start of Access Class 4	802.4
0E	2	TAR_ROT(4)	Target Rotation Time Access Class 4	802.4
10	2	START(2)	TRT at Start of Access Class 2	802.4
12	2	TAR_ROT(2)	Target Rotation Time Access Class 2	802.4
14	2	START(0)	TRT at Start of Access Class 0	802.4
16	2	TAR_ROT(0)	Target Rotation Time Access Class 0	802.4
18	2	START(RM)	TRT at Start of Ring Maintenance	802.4
1A	2	TAR_ROT(RM)	Target Rotation Time Ring Maintenance	802.4
1C	2	RM_INIT	Ring Maintenance Timer Initial Value	802.4
1E	2	SID	Bridge Pair Source Segment Number	
20	2	TID	Bridge Pair Target Segment Number	
22	2	SR_MASK	Source Routing User Mask	
24	2	MAX_INT_SOL	Maximum_Inter_Solicit_Count	802.4
26	2	RX_ERR_MASK	RX Frame Status Error Mask	
28	2	TX_STAT(6)	Status of TX Queue (6)	
2A	4	TX_HOQ6	Tx Queue Access Class 6 HOQ Pointer	
30	2	TX_STAT(4)	Status of TX Queue (4)	
32	4	TX_HOQ4	TX Queue Access Class 4 HOQ Pointer	
38	2	TX_STAT(2)	Status of TX Queue (2)	
3A	4	TX_HOQ2	TX Queue Access Class 2 HOQ Pointer	
40	2	TX_STAT(0)	Status of TX Queue (0)	
42	4	TX_HOQ0	Tx Queue Access Class 0 HOQ Pointer	
48	4	RX_EOQ6	Rx Queue Access Class 6 EOQ Pointer	
4C	4	RX_EOQ4	Rx Queue Access Clas 4 EOQ Pointer	
50	4	RX_EOQ2	RX Queue Access Class 2 EOQ Pointer	
54	4	RX_EOQ0	RX Queue Access Class 0 EOQ Pointer	
58	4	FD_POOL_PTR	Free Frame Descriptor Pointer to First FD	
5C	4	BD_POOL_PTR	Free Buffer Descriptor Pointer to First BD	
60	2	GA_MASK_LOW	Group Address Mask — Low Word	
62[1]	4	GA_MASK_MED_HI	Group Address Mask — Medium and High Words	
66	2	IA_MASK_LOW	Individual Address Mask — Low Word	
68[1]	4	IA_MASK_MED_HI	Individual Address Mask — Medium and High Words	
6C	2	MAX_RETRY	Non-RWR Maximum Retry Limit	802.4
6E	2	RWR_MAX_RETRY	RWR Maximum Retry Limit	802.4
70	2	SLOT_TIME	Slot Time Value	802.4
72	2	TS_LOW	This Station's Address — Low Word	802.4
74[1]	4	TS_MED_HI	This Station's Address — Medium and High Words	802.4
78	2	MPV_REV	Manufacturer Product Version and 802.4 Revision Number	
7A	2	TX_FLT_CNT	Transmitter Fault Count	802.4
80	2	TRT	Token Rotation Time (Current — Internal)	

NOTE:
1. This value is not meaningful if the address length is 16 bits.

Table 3-7. Set Value Opcodes

Code (Hex)	Length in Bytes	Name	Description	
08	2	HI_PRI_THT	High_Priority_Token_Hold_Time	802.4
0E	2	TAR_ROT(4)	Target Rotation Time Access Class 4	802.4
12	2	TAR_ROT(2)	Target Rotation Time Access Class 2	802.4
16	2	TAR_ROT(0)	Target Rotation Time Access Class 0	802.4
1A	2	TAR_ROT(RM)	Target Rotation Time Ring Maintenance	802.4
1C	2	RM_INIT	Ring Maintenance Timer Initial Value	802.4
1E	2	SID	Bridge Pair Source Segment Number	
20	2	TID	Bridge Pair Target Segment Number	
22	2	SR_MASK	Source Routing User Mask	
24	2	MAX_INT_SOL	Maximum_Inter_Solicit_Count	802.4
26	2	RX_ERR_MASK	RX Frame Status Error Mask	
28	2	TX_STAT(6)	Status of TX Queue (6)	
2A	4	TX_HOQ6	Tx Queue Access Class 6 HOQ Pointer	
30	2	TX_STAT(4)	Status of TX Queue (4)	
32	4	TX_HOQ4	TX Queue Access Class 4 HOQ Pointer	
38	2	TX_STAT(2)	Status of TX Queue (2)	
3A	4	TX_HOQ2	TX Queue Access Class 2 HOQ Pointer	
40	2	TX_STAT(0)	Status of TX Queue (0)	
42	4	TX_HOQ0	Tx Queue Access Class 0 HOQ Pointer	
48	4	RX_EOQ6	Rx ueue Access Class 6 EOQ Pointer	
4C	4	RX_EOQ4	Rx Queue Access Clas 4 EOQ Pointer	
50	4	RX_EOQ2	RX Queue Access Class 2 EOQ Pointer	
54	4	RX_EOQ0	RX Queue Access Class 0 EOQ Pointer	
58	4	FD_POOL_PTR	Free Frame Descriptor Pointer to First FD	
5C	4	BD_POOL_PTR	Free Buffer Descriptor Pointer to First BD	
60	2	GA_MASK_LOW	Group Address Mask — Low Word	
62[1]	4	GA_MASK_MED_HI	Group Address Mask — Medium and High Words	
66	2	IA_MASK_LOW	Individual Address Mask — Low Word	
68[1]	4	IA_MASK_MED_HI	Individual Address Mask — Medium and High Words	
6C	2	MAX_RETRY	Non-RWR Maximum Retry Limit	802.4
6E	2	RWR_MAX_RETRY	RWR Maximum Retry Limit	802.4

NOTE:
1. This value is not meaningful if the address length is 16 bits.

The command result area is the four word area following the command parameter area in the initialization table and is located at offset 96 through 9C. The first three words of the command result area are used to pass parameters from the TBC to the host while the fourth word contains the status. The least significant bit of the CPA RESULT (fourth word of the command parameter area), when set, indicates the TBC has completed the execution of the previous command. This bit, referred to as the done bit, is valid for execution of all TBC commands except for the RESET and LOAD INIT TABLE FC commands. The only confirmation for these commands is the value 'FF' of the semaphore register. If the done bit in the status word of the CPA is to be used to determine the completion of a command, it must be cleared by the host prior to giving the command. Another method to check to see if the TBC has finished processing a command is through the TBC command complete bit (TCC) in the interrupt status word 0 located in the initialization table (for all commands except RESET, LOAD INIT TABLE FC, and CLEAR INTERRUPT STATUS).

3.5.2 READ VALUE Command

The read value command returns the current operational values of TBC parameters to the host. Not all of the TBC parameters are meaningful at all times and the TBC does not indicate whether the value it returns is meaningful. The opcode of the parameter to be read is loaded into the command parameter area word 0 by the host processor. A list of these parameter opcodes is given in Table 3-6. The TBC moves the current value of the parameter into the command return field. In most cases, a two-byte value is returned in CPA RET0, and a four-byte value is returned in CPA RET0 and CPA RET1. The length of the returned value is not a parameter of the command but is a function of the parameter requested. When finished, the TBC gives confirmation of command execution in CPA RESULT.

The following exceptions exist:

1. If the opcode is '00' (next station address) or '04' (previous station address), and the address length is 48 bits, the resulting address will be returned as the high word in CPA RET0, the medium word in CPA RET1, and the low word in CPA RET2. If the address is 16 bits, the result will be returned in CPA RET2.
2. If the opcode is '80' (internal token rotation time), the value is always returned in CPA RET2. Note that this parameter is prescaled by three or six bits according to the prescaler specified via the SET MODE 3 command.
3. The boolean 'next_station_known' is returned in CPA RESULT bit 15 if the host has requested the next station address (opcode 00). If next station known is true, this bit is zero and the next station address that is returned is valid; if false, this bit is one and the next station value that is read is not meaningful.

The code for the READ VALUE command is 90 and its format is shown below.

7	6	5	4	3	2	1	0
1	0	0	1	0	0	0	0

3.5.3 SET VALUE Command

An array of commands, referred to as SET VALUE commands, are available to the user to initialize or modify the TBC's parameters. The parameters which may be modified include the function codes for buffer descriptors, frame descriptors, and receive and transmit data buffers. Also included is the capability to set the pad timer preset register (PTP), and some of the parameters which reside in the private area (see Table 3-7 for a complete list). The mechanism used to pass parameters for the SET VALUE commands is described in each case in the following subsections. The coding for the SET VALUE commands is 81 through 87 and the format for each command is shown below.

7	6	5	4	3	2	1	0
1	0	0	0	0	S	S	S

Where:

S	S	S	Function
0	0	1	Set Function Code of Buffer Descriptor[1]
0	1	0	Set Function Code of Frame Descriptor[1]
1	1	1	Set Function Code of RX and TX Data Buffers[1]
0	1	1	Set Pad Timer Preset (PTP) Register
1	1	0	Set One Word Value
1	0	1	Set Two Word Value

NOTE:
1. If the signals FC0-FC3 are not used, these commands are not necessary.

3.5.3.1 BUFFER DESCRIPTOR FUNCTION CODE. This command sets a single function code used for all buffer descriptors. The host must write the desired function code (FC0-FC3) into bits 0-3 of CPA VAL0 before loading the command register with this command. This command is normally given only as part of an initialization sequence or during testing. The coding is 81 and the format is shown below:

7	6	5	4	3	2	1	0
1	0	0	0	0	0	0	1

3.5.3.2 FRAME DESCRIPTOR FUNCTION CODE. This command sets a single function code used for all frame descriptors. The host must write the desired function code (FC0-FC3) into bits 0-3 of CPA VAL0 before loading the command register with this command. This command is normally given only as part of an initialization sequence or in testing. The coding is 82 and the format is shown below:

7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	0

3.5.3.3 DATA BUFFERS FUNCTION CODE. This command sets a single function code used for all receive and transmit data buffers. The host must write the desired function code (FC0-FC3) into bits 0-3 of CPA VAL0 before loading the command register with this command. This command is normally given only as part of an initialization sequence or during testing. The coding is 87 and the format is shown below:

7	6	5	4	3	2	1	0
1	0	0	0	0	1	1	1

3.5.3.4 SET PAD TIMER PRESET REGISTER (PTP). The pad timer preset register is used by the TBC to set the length and pattern of the preamble, and the minimum number of preamble octets transmitted between frames. The initial value of this register is loaded by the host at offset 78 of the initialization table. After initialization this command must be used to modify the PTP. The host must write the new PTP value into bits 0-15 of CPA VAL0 before loading the command register with this command. This register contains three parameters which include:

1. The preamble pattern, in bits 0-7 where bit 0 is the first bit of each preamble octet transmitted and bit 7 is the last.
2. The minimum number of preamble octets transmitted after silence is loaded in bits 8-10. This is equivalent to the 802.4 parameter min_post_silence_preamble length.
3. The minimum number of preamble octets transmitted between frames resides in bits 11-15.

The coding for the SET PTP command is 83 and its format is:

7	6	5	4	3	2	1	0
1	0	0	0	0	0	1	1

The format of the pad timer preset (PTP) register is:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
m	m	m	m	m	b	b	b	p	p	p	p	p	p	p	p

Where:

m = Minimum number (minus 1) of preamble octets transmitted between frames.
b = Minimum number (minus 2) of preamble octets transmitted before frame after silence.
p = Pattern of preamble octet.

Minimum values for the PTP register are:

Physical Layer Type	Bit Rate (Mb/Sec)	m	m	m	m	m	b	b	b	p	p	p	p	p	p	p	p
Single Channel Phase Continuous FSK	1	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1	0
Single Channel Phase Coherent FSK	5	0	0	0	0	1	1	1	1	1	0	1	0	1	0	1	0
	10	0	0	0	1	0	1	1	1	1	0	1	0	1	0	1	0
Broadband Bus	1	0	0	0	0	0	0	1	0	1	0	1	0	1	0	1	0
	5	0	0	0	0	1	0	1	0	1	0	1	0	1	0	1	0
	10	0	0	0	1	0	0	1	0	1	0	1	0	1	0	1	0

The values for 'mmmmm' and 'bbb' in the above table are minimum values. The user may use larger values for 'mmmmm' to enhance back-to-back frame reception, or may use larger values for 'bbb' if a non-standard modem being used requires more preamble after silence.

The PTP register bits have different meanings when being used in the receive test. See **3.6.5 RECEIVER TEST Command** for more details.

3.5.3.5 SET ONE WORD. The SET ONE WORD command is used to set two-byte TBC operational parameters residing in the private area. The host must load the opcode into CPA VAL0 and the new parameter value in CPA VAL1 before writing the SET ONE WORD command into the command register. The opcodes for the SET ONE WORD/TWO WORDS commands are listed in Table 3-7. The coding for this command is 86 and its format is shown below:

7	6	5	4	3	2	1	0
1	0	0	0	0	1	1	0

3.5.3.6 SET TWO WORDS. The SET TWO WORDS command is used to set four-byte TBC parameters which reside in the private area. The host must load the opcode into CPA VAL0 and the new parameter value high order word in CPA VAL1 and low order word in CPA VAL2 before writing the SET TWO WORDS command into the command register. The coding for this command is 85 and its format is shown below:

7	6	5	4	3	2	1	0
1	0	0	0	0	1	0	1

The opcodes for the SET ONE WORD/TWO WORDS commands are listed in Table 3-7.

3.6 TESTING

Commands in this category were designed to facilitate debugging and diagnostics tasks on a TBC system. There are four available tests which can be run on the TBC while in the OFFLINE state. The receiver test, transmitter test, and full duplex loopback test may be run in either internal or external loopback mode, see **3.6.1 SET INTERNAL/EXTERNAL LOOPBACK MODE Command** for a description. The codings for the test commands are B0 through BC and the format is:

7	6	5	4	3	2	1	0
1	0	1	1	T	T	T	T

Where:

T	T	T	T	
0	0	0	0	Receiver test
0	1	sil/noi	0	Transmitter test
1	0	0	0	Full duplex loopback test
1	1	0	0	Host interface test

The host interface test exercises the DMA portion of the TBC by checking the path from a memory buffer to the TBC and back to the memory buffer. The transmitter test is used to check the path from a transmit queue in memory through the transmitter and back to the receiver. The receiver test is used to check the path from the receiver to a receive queue in memory. Finally, the full duplex loopback test is used to check the path from a buffer in memory through the TBC serial interface and back to the memory buffer.

3.6.1 SET INTERNAL/EXTERNAL LOOPBACK MODE Command

The SET INTERNAL/EXTERNAL LOOPBACK MODE command is used by the host processor to enable or disable the internal loopback mode of operation while running the receiver, transmitter, and full duplex loopback tests. Note that if internal loopback mode is selected, an external clock has to be provided for the receive clock while running the tests. The default mode of operation is with internal loopback mode disabled. The coding for this command is CC or C4 and its format is shown below.

7	6	5	4	3	2	1	0
1	1	0	0	INTERNAL LOOPBACK	1	0	0

INTERNAL LOOPBACK

1 Enables internal loopback mode of operation.

0 Disables internal loopback mode of operation.

This command controls the internal loopback from TXSYM0, TXSYM1, and TXSYM2 to RXSYM0, RXSYM1, and RXSYM2 respectively. In internal loopback mode, transmitted information is looped back as received information internally to the chip. Also, in internal loopback mode, the external transmit output pins send silence symbols regardless of the symbols being generated internally. In external loopback mode, the transmit pins must be externally connected to the receive pins either directly or through a modem.

3.6.2 HOST INTERFACE TEST Command

The host interface test is used to check the path from the memory buffer to the TBC FIFO and back to the memory buffer. To initiate this test, the user first builds a 70-byte buffer placing test data in the first 34 bytes, see Table 3-8 for format. Next, the user loads the function code of the buffer into CPA VAL0 of the command parameter area, loads the buffer pointer into CPA VAL1 (high word) and CPA VAL2 (low word) and issues the test command with the appropriate bits set for the host interface test. The 34 bytes of data are copied from the first half of the 70-byte buffer to the TBC and written back to the second half of the 70-byte buffer by the TBC. Upon completion of the test, the done bit in the status word of the CPA located at offset 9C in the initialization table is set. The host checks the data written back to the memory buffer against what was transferred.

Table 3-8. Test Buffer Format

Buffer Format:

MSB F E D C B A 9 8	LSB 7 6 5 4 3 2 1 0
Host Data 0	Host Data 1
Host Data 2	Host Data 3
. . .	. . .
. . .	. . .
Host Data 32	Host Data 33
TBC Return Data 0	TBC Return Data 1
TBC Return Data 2	TBC Return Data 3
. . .	. . .
. . .	. . .
TBC Return Data 32	TBC Return Data 33
Indication 0	Indication 1

Host Data:
Data prepared by the host which resides in the first 34 bytes.

TBC Return Data:
Data returned by the TBC which resides in the next 34 bytes and which should be the same as the host data.

Indication:
Indication returned by the TBC in full duplex loopback test. This value is 0 in the host interface test.

The following routine may be used to perform the host interface test:

Prepare the Initialization Table
Issue RESET command
Wait until Semaphore Register is 'FF'
Initialize the Interrupt Vector Register
Load the Function Code of the init table pointer into DR0
Issue LOAD INITIALIZATION TABLE FC Command
Wait until Semaphore Register is 'FF'
Load the Init Table Pointer into DR
Issue INITIALIZE Command
Issue SET MODE 3 Command to set SWAP and HLEN Bits
Prepare First 34 Bytes of Test Buffer
Load CPA VAL0 with Function Code of Buffer if Needed
Load CPA VAL1 and CPA VAL2 with Pointer to Buffer
Clear Done Bit in CPA Status Word
Issue HOST INTERFACE TEST (Code is BC)
Wait for Done Bit in CPA Status Word
Compare Returned Data to Data which was given to the TBC

NOTE

In order to check for command completion, the host must clear the done bit in the CPA status word before issuing a command to the TBC.

3.6.3 FULL-DUPLEX LOOPBACK TEST Command

The full-duplex loopback test checks the path from buffer memory through the TBC serial interface and back to the memory buffer, thus checking the proper functioning of the transmit and receive

machines. Both the receive and transmit machines are working simultaneously, i.e., in full duplex, while running this test. To run this test, the user must first prepare a buffer as described in Table 3-8. The buffer is used in the same way as for the host interface test with the following exceptions:

1. Byte 0 (host data 0) is not transmitted.
2. Byte 34 (TBC ret data 0) does not contain valid information.
3. The buffer length may vary according to the settings of mode bits for RCDS, RX CRC to data storage, and TCDS, TX CRC disable, (see **3.3.3 SET MODE 3 Command**). Table 3-9 shows the buffer length relationship with these mode bits.
4. Indication 0 byte, that is the first byte after the returned data, contains the frame data length including the CRC. Note that this value is valid only if no CRC error was detected.
5. The most significant bit, bit 7, of indication 1 byte is set if a CRC error was detected by the TBC.

Table 3-9. Test Buffer for Returned Data

RCDS	TCDS	Data Length Stored in Buffer	Indication 1 (Error) Byte Number
0	0	33	69
0	1	29	65
1	0	37	73
1	1	33	69

3

Note that the indication bytes do not contain sufficient information to determine whether the test was a success or failure. The data itself must be checked by the host.

The following routine may be used to run the full duplex loopback test:

Prepare the Initialization Table
Issue RESET Command
Wait until Semaphore Register is 'FF'
Initialize the Interrupt Vector Register
Load the Function Code of the Init Table Pointer into DR0
Issue LOAD INITIALIZATION TABLE FC Command
Wait until Semaphore Register is 'FF'
Load the Init Table Pointer into DR
Issue INITIALIZE Command
Issue SET MODE 3 Command to Set SWAP, HLEN, RCDS and TCDS Bits
Prepare Test Buffer
Load CPA VAL0 with Function Code of Buffer if needed
Load CPA VAL1 and CPA VAL2 with Pointer to Buffer
Issue SET INTERNAL/EXTERNAL LOOPBACK MODE
Initialize Modem if in External Loopback Mode
Clear Done Bit in CPA Status Word
Issue FULL DUPLEX LOOPBACK MODE TEST (Code is B8)
Wait for Done Bit in Status Word
Read Indication 0 and 1 for Errors
Compare Returned Data to Data which was given to the TBC

NOTE

In order to check for command completion, the host must clear the done bit in CPA status word before issuing a command to the TBC.

3.6.4 TRANSMITTER TEST Command

The transmitter test is used by the host processor to check the path from the TX queue in memory to the transmitter and back to the receiver. During this test, the TBC acts as if it is in_ring and has just received a token. The transmitter test may be run with the option of waiting for non-silence (bit 1 of the command) on the bus before transmission begins, thus deliberately creating a collision. Note that if this test is run in internal loopback mode, it must also be run with the silence/noise bit set to one, that is the TBC will wait for reported silence to start transmitting. If desired, a cyclic transmit queue can be used, creating an almost unlimited number of frames for transmission (see **SECTION 4 BUFFER STRUCTURES**). The number of frames is only limited by the high_priority_token_hold time when in external loopback mode, while there are no limitations in internal loopback mode. During the transmitter test, the receiver is in a special mode in which it checks incoming frames for CRC (FCS) and underrun errors but does not write the contents of the frame to the FIFO. To perform this test, the host must prepare a class 6 transmission queue and set the high_priority_token_hold_time to 'FFFF' for maximum transmitter test time. When either the last transmission frame has been confirmed, or an interrupt on empty transmission queue has occurred (bit three of interrupt status word 0, TXQE), the test should be ended by issuing the OFFLINE command. Command confirmation and status are returned to the command return area in the initialization table. The OFFLINE command must be given before the token hold timer expires while running in external loopback mode. To check the tests results, the host can check the FD confirmation word for TX queue access class 6. If detected, CRC/underrun errors are indicated by the TBC setting bit 1 of the command result area after confirmation was given for the OFFLINE command.

The following routine may be used to perform the transmitter test:

Prepare the Initialization Table
Issue RESET Command
Wait until Semaphore Register is 'FF'
Initialize the Interrupt Vector Register
Load the Function Code of the Init Table Pointer into DR0
Issue LOAD INITIALIZATION TABLE FC Command
Wait until Semaphore Register is 'FF'
Load the Init Table Pointer into DR
Issue INITIALIZE Command
Issue SET MODE 3 Command to Set SWAP, HLEN, PS3 and TCDS Bits
Prepare Access Class 6 TX Queue
Issue SET ONE WORD Command to Set High_Priority_Token_Hold_Time if Different from Value Set in Initialization Table
Issue SET FC BD, FD, and RX/TX Data Buffers if needed
Issue SET INTERNAL/EXTERNAL LOOPBACK MODE
Initialize Modem if in External Loopback Mode
Issue START Command for Access Class 6
Clear Done Bit in CPA Status Word
Issue TRANSMITTER TEST (Code is B4 or B6)
Wait for Command Confirmation which Indicates Acceptance of the Test
Wait until Last Known TX Frame has been Confirmed or an Interrupt has been Generated by the TBC for TXQE

Issue the OFFLINE Command

Check for CRC/Underrun Errors (Bit 1 of the Command Result area will be set if such errors occurred.)

Check the FD Confirmation Word for Each Transmitted Frame

NOTE

In order to check for command completion, the host must clear the done bit in CPA status word before issuing a command to the TBC.

3.6.5 RECEIVER TEST Command

The receiver test is used by the host processor to check the path from the receiver to the RX queue in memory. The transmitter transmits a predefined pattern partitioned into frames of up to 33 bytes in length. The pattern of the frame is user definable and is loaded by the host through CPA VAL0. This pattern will be transmitted between the SD and FCS (CRC) of frames from the least significant byte to the most significant byte. However, care should be taken not to create an RWR pattern. For this test to run, it is required that the TBC generate the CRC, therefore the TCDS bit should be left disabled using the SET MODE 3 command (see **3.3.3 SET MODE 3 Command**). The PTP register has a different format in this test than in normal mode as shown.

PTP register format for the receive test:

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
m	m	m	m	m	b	b	b	p	p	p	p	p	p	p	p

Where:

m = Frame's length - number of bytes between the start delimiter up to but not including the CRC (FCS) minus two (2).

b = Number of preamble octets between frames minus one (1).

p = Pattern of preamble octet

The command confirmation bit is set by the TBC upon completion of the OFFLINE command and the status is placed in the command return area. Status is given on every frame in the FD (class 6) as in normal frame reception. In order to determine success or failure of the test, the host processor checks the received frames against what was sent in CPA VAL0.

NOTES

Using a 1:1 ratio of the serial clock to the system clock, an overflow is expected after approximately every third frame. To reduce the number of overruns when running this test, it is recommended to use 16 words as the minimum data buffer length, and to allocate one buffer descriptor per frame descriptor.

The following routine may be used to perform the receiver test:

Prepare the Initialization Table

Issue RESET Command

Wait until Semaphore Register is 'FF'

Initialize the Interrupt Vector Register

Load the Function Code of the Init Table Pointer into DR0

Issue LOAD INITIALIZATION TABLE FC Command

Wait until Semaphore Register is 'FF'

Load the Init Table Pointer into DR

Issue INITIALIZE Command

Issue SET MODE 1 Command to Set CACF and CUFC Mode Bits

Issue SET MODE 3 Command to Set SWAP, HLEN, PS3, and RCDS Bits if Desired

Issue SET 1 WORD or 2 WORDS Command to Set Individual Address Mask to Zero (Copy all Frames)

Issue SET 1 WORD or 2 WORDS Command to Set Group Address Mask to 'FFFF' (Copy all Frames)

Issue SET PTP Command

Prepare Free FD and BD Pools

Issue SET 2 WORDS Commands to Initialize the Free FD and BD Pool Pointers as well as the RX Queue Access Class 6 EOQ Pointer

Issue SET FC BD, FD, and RX/TX Data Buffers if needed

Issue SET INTERNAL/EXTERNAL LOOPBACK MODE

Initialize Modem if in External Loopback Mode

Load CPA VAL0 with Pattern Word — Must be Two Identical Bytes

Clear Done Bit in CPA Status Word

Issue RECEIVER TEST (Code is B0)

Wait for Command Confirmation which Indicates Acceptance of the Test

Wait until FD or BD Pool Empty (Bits 1 and 2 in Interrupt Status Word 0)

Issue the OFFLINE Command

Check for Received Frames' Status in the FD (Class 6)

NOTE

In order to check for command completion, the host must clear the done bit in CPA status word before issuing a command to the TBC.

3.6.6 Sequence for Running all the Tests

The tests may be run one after the other without reinitialization, provided each preceding test runs successfully to completion. If a test fails, it is strongly recommended to reinitialize the chip before running the next test. This sequence of tests can be run at system initialization and after reset. The four tests are executed in the following sequence: host interface, full duplex loopback, receive, and transmit. This routine should be performed as follows:

TBC INITIALIZATION

Prepare the Initialization Table

Issue RESET Command

Wait until Semaphore Register is 'FF'

Initialize the Interrupt Vector Register

Load the Function Code of the Init Table Pointer into DR0

Issue LOAD INITIALIZATION TABLE FC Command
Wait until Semaphore Register is 'FF'
Load the Init Table Pointer into DR
Issue INITIALIZE Command

HOST INTERFACE TEST

Issue SET MODE 3 Command to Set SWAP and HLEN Bits
Prepare First 34 Bytes of Test Buffer
Load CPA VAL0 with Function Code of Buffer if Needed
Load CPA VAL1 and CPA VAL2 with Pointer to Buffer
Clear Done Bit in CPA Status Word
Issue HOST INTERFACE TEST (Code is BC)
Wait for Done Bit in Status Word
Compare Returned Data to Data which was given to the TBC

FULL DUPLEX LOOPBACK TEST

Issue SET MODE 3 Command to Set SWAP, HLEN, RCDS, and TCDS Bits if Needed
Prepare Test Buffer
Load CPA VAL0 with Function Code of Buffer if needed
Load CPA VAL1 and CPA VAL2 with Pointer to Buffer
Issue SET INTERNAL/EXTERNAL LOOPBACK MODE
Initialize Modem if in External Loopback Mode
Clear Done Bit in CPA Status Word
Issue FULL DUPLEX LOOPBACK MODE TEST (Code is B8)
Wait for Done Bit in Status Word
Read Indication 0 and 1 for Errors
Compare Returned Data to Data which was given to the TBC

TRANSMITTER TEST

Issue SET MODE 3 Command to set SWAP, HLEN, PS3, and TCDS Bits if Needed
Prepare Access Class 6 TX Queue
Issue SET ONE WORD Command to Set High_Priority_Token_Hold Time if Different from Value Set in Initialization Table
Issue SET FC BD, FD, and RX/TX Data Buffers if Needed
Issue SET INTERNAL/EXTERNAL LOOPBACK MODE
Initialize Modem if in External Loopback Mode Needed
Issue START Command for Access Class 6
Clear Done Bit in CPA Status Word
Issue TRANSMITTER TEST (Code is B4 or B6)
Wait for Command Confirmation which Indicates Acceptance of the Test
Wait until Last Known TX Frame has been Confirmed or an Interrupt is Generated by the TBC for TXQE
Issue the OFFLINE Command

Check for CRC/Underrun Errors (Bit 1 of the command result area will be set if such errors occurred.)

Check the FD Confirmation Word for each Transmitted Frame

RECEIVER TEST

Issue SET MODE 1 Command to set CACF and CUFC Mode Bits

Issue SET MODE 3 Command to set SWAP, HLEN, PS3, and RCDS Bits if Desired

Issue SET 1 WORD or 2 WORDS Command to Set Individual Address Mask to Zero (Copy all Frames)

Issue SET 1 WORD or 2 WORDS Command to Set Group Address Mask to 'FFFF' (Copy all Frames)

Issue SET PTP Command

Prepare Free FD and BD Pools

Issue SET 2 WORDS Commands to Initialize the Free FD and BD Pool Pointers as well as the RX Queue Access Class 6 EOQ Pointer

Issue SET FC BD, FD, and RX/TX Data Buffers if Needed

Issue SET INTERNAL/EXTERNAL LOOPBACK MODE

Initialize Modem if Needed

Load CPA VAL0 with Pattern Word - MUST be Two Identical Bytes

Clear Done Bit in CPA Status Word

Issue RECEIVER TEST (Code is B0)

Wait for Command Confirmation which Indicates Acceptance of the Test

Wait until FD or BD Pool Empty (Bits 1 and 2 in Interrupt Status Word 0)

Issue the OFFLINE Command

Check for Received Frames' Status in the FD (Class 6)

NOTE

In order to check for command completion, the host must clear the done bit in CPA status word before issuing a command to the TBC.

3.7 NOTIFY TBC

The two commands in this category are used to notify the TBC that a response frame is ready when using the non-predefined RWR mechanism or to clear some interrupt status bits.

3.7.1 CLEAR INTERRUPT STATUS Command

This command is used to clear, that is negate, the interrupt request signal ($\overline{IRQ}$) and specific status bits in interrupt status words 0 and 1 which are located in the initialization table (see **2.2.11 Interrupt Status Words**). Before loading this command into the command register, the host must write the mask for interrupt status word 0 into CPA VAL0 while the mask for interrupt status word 1 must be loaded into CPA VAL1. The mask consists of a '0' to leave the corresponding status bit unchanged and a '1' to cause it to be cleared. After the TBC has cleared the desired bits, if there remains a set, unmasked status bit in one of the status words, interrupt request will be reasserted even if it was negated by the previous CLEAR INTERRUPT STATUS command. Upon receiving this command the TBC will immediately negate $\overline{IRQ}$. The status bits in the initialization table will

be cleared only when the TBC actually executes the command. A situation may arise in which the CPU may get an interrupt without any status bits in the interrupt status words being set. This scenario may occur if the host happens to be issuing a CLEAR INTERRUPT STATUS command right after the TBC has updated one of the interrupt status words for a new event but before the TBC generated an $\overline{IRQ}$. If that situation occurs the host should issue the CLEAR INTERRUPT STATUS COMMAND with a zero mask in CPA VAL0 and CPA VAL1. Note that the TBC command complete bit (TCC) residing in interrupt status word 0 is not set after completion of this command.

The coding of this command is 88 and its format is shown below:

7	6	5	4	3	2	1	0
1	0	0	0	1	0	0	0

3.7.2 RESPONSE READY Command

This command is used by the host to notify the TBC that a response frame is ready to be transmitted following the reception of a request with response frame. The issuance of this command is an indication to the TBC that the response pointer in the initialization table located at offset 88 is valid. This command is only applicable when not in predefined response mode, see **4.5 RECEPTION OF RWR FRAMES AND TRANSMISSION OF RESPONSE** for a description of this mechanism.

The coding for this command is C5 and its format is shown below:

7	6	5	4	3	2	1	0
1	1	0	0	0	1	0	1

3.8 MODEM CONTROL

Two commands are provided to allow the TBC to communicate management services to the physical layer of the network.

3.8.1 PHYSICAL Command

This command is used by the host processor to control the modem while in station management and should only be given when the TBC is in the OFFLINE state. Status is returned to the last word in the command parameter area (CPA RESULT) as shown following the command format. The physical data request channel is fully described in **5.9.1 Physical Data Request Channel** while the physical data indication channel is described in **5.9.2 Physical Data Indication Channel.** Note that $\overline{SMREQ}$ equals zero is an indication that station management has been selected. This signal is automatically asserted when the first physical command is issued by the host. $\overline{SMREQ}$ will stay low until the END PHYSICAL command (see **3.8.2 END PHYSICAL Command**) is issued by the host to terminate station management. $\overline{SMREQ}$ is also referred as TXSYM3 in the 802.4G document.

The coding for the physical command is A1-A7 and its format is shown below:

7	6	5	4	3	2	1	0
1	0	1	0	0	DATA	TXSYM2	TXSYM1

TXSYM2
- 1 TXSYM2 line to the modem is set to one while $\overline{\text{SMREQ}}$ equals zero
- 0 TXSYM2 line to the modem is set to zero while $\overline{\text{SMREQ}}$ equals zero

TXSYM1
- 1 TXSYM1 line to the modem is set to one while $\overline{\text{SMREQ}}$ equals zero
- 0 TXSYM1 line to the modem is set to zero while $\overline{\text{SMREQ}}$ equals zero

DATA
- 1 Perform data exchange with the modem
- 0 No data exchange

Note that the PHYSICAL command should not be given with a code of A0 as this commands the modem to enter the idle state without a data transfer. This action would then cause the modem to respond with idle whereas the TBC would be waiting for ACK or NACK. Only the TBC is permitted to issue idle to the modem without a data transfer.

The status word format returned in CPA RESULT is shown below:

7	6	5	4	3	2	1	0
X	X	X	X	X	RXACK	RXNA	COMCON

RXACK — RXSYM1 Acknowledgement
- 1 Acknowledgement received
- 0 No acknowledgement received

RXNA — RXSYM2 Non-Acknowledgement
- 1 Non-acknowledged received
- 0 No non-acknowledgement received

COMCON — Command Configuration
- 1 Command confirmed
- 0 Command not confirmed

X - Don't Care

There are two main types of commands which can be issued by the host to the physical layer: immediate and data transfer. Immediate commands do not involve data exchange with the modem but rather provide a simple interface to allow the host to control the modem through the TBC. Data transfer commands are designed for intelligent modems which need additional information to be transferred between the host and the modem through the TBC.

3.8.1.1 IMMEDIATE COMMANDS. The immediate commands include reset, disable loopback, and enable transmitter. Refer to **5.9.1.4 TXSYM2, TXSYM1, AND TXYSM0 IN STATION MANAGEMENT MODE** for a description of the encodings of these commands on the physical interface. The following paragraphs describe the handshake between the TBC and the modem. The host loads the command into the command register and the TBC passes the command to the modem by encoding it on pins TXSYM1 and TXSYM2 while TXSYM0 is set to one. Note that for immediate commands, bit 2 (DATA) of the command register must be set to zero. Upon receiving this type of command the TBC will go through the steps described below:

1. It is assumed that the TBC and the modem are in MAC mode or that a modem error just occurred.

2. The TBC encodes the command found in CR on TXSYM1 and TXSYM2 with TXSYM0=1 and $\overline{\text{SMREQ}}$ equal to zero.
3. The TBC waits indefinitely for ACK or NACK coding on RXSYM1 and RXSYM2 as well as $\overline{\text{SMIND}}$=0 from the modem.

NOTES

The modem is free to go to idle before transitioning to this state. If the modem does not respond after a reasonable length of time, a software or hardware RESET must be issued to the TBC.

4. The TBC samples the ACK/NACK signals (RXSYM1 and RXSYM2) and issues idle to the modem (TXSYM2=0, TXSYM1=0, while TXSYM0=1, and $\overline{\text{SMREQ}}$=0).
5. The TBC waits for the modem to respond with idle (RXSYM2=0, RXSYM1=0, while $\overline{\text{SMIND}}$=0).

NOTE

If the modem does not respond after a reasonable length of time, a software or hardware RESET must be issued to the TBC.

6. The TBC updates the CPA result with the ACK/NACK and command done bits.
7. At this point, the TBC is in SM idle mode with TXSYM2=0, TXSYM1=0, TXSYM0=1, and $\overline{\text{SMREQ}}$=0. The modem is in the idle state with RXSYM2=0, RXSYM1=0, and $\overline{\text{SMIND}}$=0.
8. The TBC is ready to accept another command.

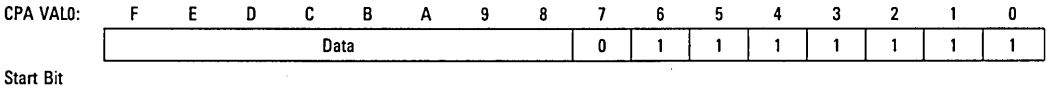
3.8.1.2 DATA TRANSFER COMMANDS. Data transfer commands are designed for intelligent modems which need more information. The serial station management data interface provides a sophisticated way for the host to communicate with the modem through the TBC. Upon receiving this type of command the TBC will go through the steps described below:

1. It is assumed that the TBC and the modem are in MAC mode or that a modem error just occurred.
2. The TBC issues the data transfer command found in CR with TXSYM0=1 and $\overline{\text{SMREQ}}$=0 to the modem.
3. The TBC waits indefinitely for the modem to assert $\overline{\text{SMIND}}$=0.

NOTE

If the modem does not respond after a reasonable length of time, a software or hardware RESET must be issued to the TBC.

4. The TBC samples RXSYM1 and RXSYM2 for ACK/NACK from now to the end of the received data stream.
5. The TBC outputs the bit stream taken from CPA VAL0 (16 bits from 0 to 15) on TXSYM0. The data word in CPA VAL0 must include the start bit as shown to conform to 802.4G:



NOTE

The programmer does not have to furnish the stop bit.

To conform to 802.4G only one data byte may be sent at a time. If more than one data byte needs to be sent, then more data commands must be given.

6. The TBC waits indefinitely for the modem to assert the start bit i.e., RXSYM0 = 0.

NOTE

If the modem does not respond after a reasonable length of time, a software or hardware RESET must be issued to the TBC.

7. The TBC waits 16 clock cycles to sample the 16 bits after the start bit from RXSYM0.
8. The TBC asserts TXSYM0 = 1, TXSYM1 = 0, TXSYM2 = 0 with $\overline{\text{SMREQ}}$ = 0, thus issuing idle.
9. The TBC waits for the modem to respond with idle: RXSYM1 = 0, RXSYM2 = 0, while $\overline{\text{SMIND}}$ = 0.
10. The TBC then copies the received word into CPA RET 0. Bits 0 through 7 contain the data returned by the modem while bits 8 through F contain all ones if the 802.4G interface was used.
11. The TBC updates the CPA result with the ACK/NACK and command done bits.
12. At this point, the TBC is in SM idle mode with TXSYM2 = 0, TXSYM1 = 0, TXSYM0 = 1, and $\overline{\text{SMREQ}}$ = 0. The modem is in the idle state with RXSYM2 = 0, RXSYM1 = 0, and $\overline{\text{SMIND}}$ = 0.
13. The TBC is ready to accept another command.

3.8.2 END PHYSICAL Command

This command causes the TBC to exit from station management mode. Upon receiving the command, the TBC sets $\overline{\text{SMREQ}}$ = 1, and waits for the modem to set $\overline{\text{SMIND}}$ = 1 before setting the command done bit (bit 0 of CPA RESULT). The coding for this command is A8 and its format is:

7	6	5	4	3	2	1	0
1	0	1	0	1	0	0	0

3.9 ILLEGAL COMMAND

Commands with bits 5, 6, and 7 set to zero are ILLEGAL commands to the TBC. A command with this format will have no affect on the TBC and the command complete bit in the status word will not be set. The format of an ILLEGAL command is shown below. The semaphore register will contain 'FE' until the next command is passed by the host processor to the TBC.

7	6	5	4	3	2	1	0
0	0	0	X	X	X	X	X

X — Don't Care

SECTION 4
BUFFER STRUCTURES

The TBC communicates with the host processor primarily through shared memory. This shared memory is divided into the initialization table and buffer structures. The fully linked buffer structures include frame descriptors (FD), buffer descriptors (BD), and data buffers (DB). One frame descriptor is required per received or transmitted frame. Each frame descriptor contains information pertaining both to the frame sent or received plus a pointer to the next FD as well as a pointer to its first BD. Buffer descriptors contain the pointer to a data buffer as well as that buffer's attributes, and a pointer to the next buffer descriptor in the frame if used. The data buffers are used to store message data, and there is one data buffer associated with each buffer descriptor. Figure 4-1 illustrates the linked buffer structure.

To fully support the IEEE 802.4 message priorities, the TBC provides four transmit queues and four receive queues. Before transmission of a message, the host processor creates frame descriptors, buffer descriptors, and data buffers for that message and then links the frame descriptor to the appropriate transmit queue. Transmission queues may be enabled or disabled using the SET ONE WORD command. The TBC confirms transmission of the frames in each frame descriptor as they are sent out. During reception, the TBC reverses the process to use frame descriptors and buffer descriptors from the pre-linked free frame descriptors pool and free buffer descriptors pool as frames are received, assigning these frames to the proper reception queue. Receive queues may not be disabled. The free frame descriptor and buffer descriptor pool pointers, which are located in the private area, must be valid at initialization time and may be changed thereafter using the SET TWO WORDS command while the TBC is OFFLINE. Finally, the host processor removes the frame data from these reception queues as programmed. Figure 4-2 illustrates the linking between the queues, FDs, BDs, and DBs. Refer to **4.2 TRANSMISSION OF A FRAME** and **4.3 RECEPTION OF FRAMES** for detailed descriptions of the tranmission and reception mechanisms.

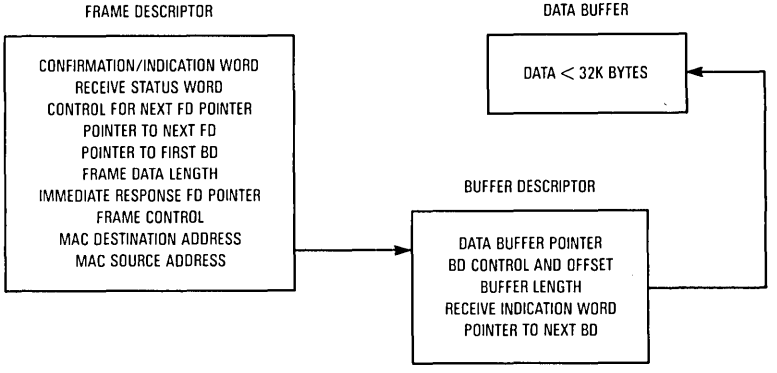


Figure 4-1. Linked Buffer Structures

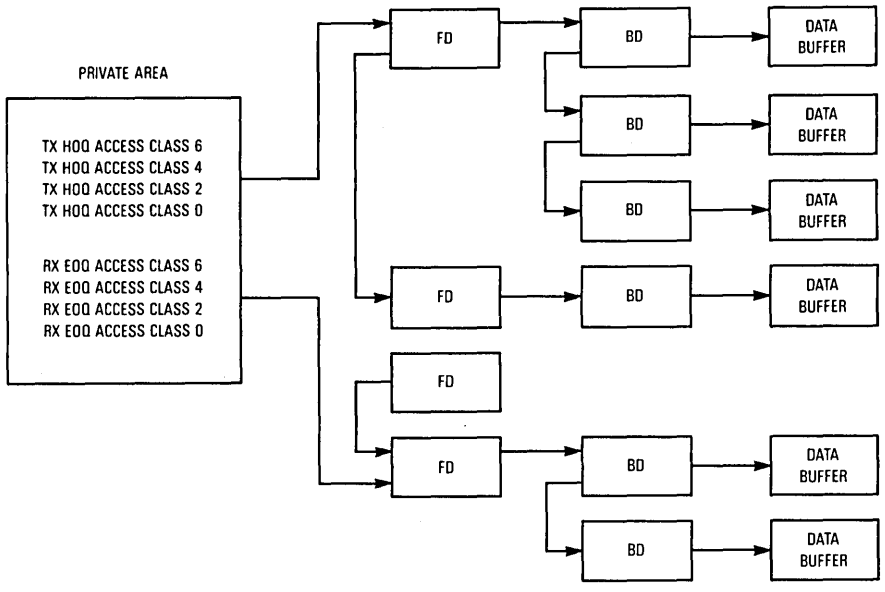


Figure 4-2. TBC Queues

The buffer management structure is powerful and very flexible. The host processor has to maintain only the free frame descriptor pool and free buffer descriptor pool for receive messages since the TBC can dynamicaily allocate buffers to the appropriate queue from the free pool. The TBC data buffer structure easily interfaces to upper layer software. An offset in the data buffer descriptor indicates how far from the beginning of the data buffer actual data begins. This offset is useful for building data frames as they are passed down through the seven layers of the ISO/OSI structure. Address and control information can then be appended to and removed from the front of each frame as it passes through the software layers during transmission and reception respectively. The TBC can access the first data buffer on an odd byte boundary while transmitting, further removing restrictions on building frames.

4.1 BUFFER STRUCTURES

The following paragraphs present in detail the TBC's buffer structures which consist of frame descriptors, buffer descriptors, and data buffers as shown in Figure 4-1.

4.1.1 Frame Descriptors

Frame descriptors (FD) contain MAC frame parameters, frame status, and pointers. Frame descriptors are used for both transmission (TX) and reception (RX) of message frames. The formats of the frame descriptors for a 48-bit MAC address and for a 16-bit MAC address are shown in Figure 4-3. Note that a frame descriptor must start at an even address.

DISPLACEMENT (IN BYTES)	DESCRIPTION OF FIELD	
0	CONFIRMATION/INDICATION WORD	
2	RECEIVE STATUS WORD	
4	CONTROL FOR NEXT FD POINTER	
6	NEXT FD POINTER — HIGH	
8	NEXT FD POINTER — LOW	
A	RESERVED	
C	FIRST BD POINTER — HIGH	
E	FIRST BD POINTER — LOW	
10	FRAME DATA LENGTH	
12	RESERVED	
14	RESERVED	
16	IR FRAME DESCIRPTOR POINTER — HIGH	
18	IR FRAME DESCRIPTOR POINTER — LOW	
1A	RESERVED	
1B	FRAME CONTROL — BYTE	
	IF ALEN = 0 THEN THE ADDRESS LENGTH IS 2 BYTES	
1C	MAC DESTINATION ADDRESS — LEAST SIGNIFICANT BYTE	
1D	MAC DESTINATION ADDRESS — MOST SIGNIFICANT BYTE	
1E	MAC SOURCE ADDRESS — LEAST SIGNIFICANT BYTE	
1F	MAC SOURCE ADDRESS — MOST SIGNIFICANT BYTE	
20	RESERVED	
22	RESERVED	
	IF ALEN = 1 THEN THE ADDRESS LENGTH IS 6 BYTES	
1C	MAC DESTINATION ADDRESS - BYTE 0	IEEE LSB
1D	MAC DESTINATION ADDRESS - BYTE 1	
1E	MAC DESTINATION ADDRESS - BYTE 2	
1F	MAC DESTINATION ADDRESS - BYTE 3	
20	MAC DESTINATION ADDRESS - BYTE 4	
21	MAC DESTINATION ADDRESS - BYTE 5	
22	MAC SOURCE ADDRESS - BYTE 0	
23	MAC SOURCE ADDRESS - BYTE 1	IEEE LSB
24	MAC SOURCE ADDRESS - BYTE 2	
25	MAC SOURCE ADDRESS - BYTE 3	
26	MAC SOURCE ADDRESS - BYTE 4	
27	MAC SOURCE ADDRESS - BYTE 5	
28	RESERVED	
2A	RESERVED	

Figure 4-3. Frame Descriptor Format

4.1.1.1 FD CONFIRMATION/INDICATION WORD FORMAT. The first word of the frame descriptor contains the frame descriptor confirmation/indication word. This word holds status information of the frame and the queue. The frame descriptor confirmation word must be cleared by the host before being placed into the free frame descriptor pool or the transmit queue. This word is used in both transmission and reception queues and is updated by the TBC.

The format of this word for a TRANSMISSION queue is shown below:

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
CFD	N/P	EMP	FTL	RES	UR	NR	UF	RT7	RT6	RT5	RT4	RT3	RT2	RT1	RT0

CFD — Confirm Frame Descriptor

0 Frame has not yet been processed (remaining indication bits are meaningless in this word)

1 TBC completed processing of this frame

N/P — Negative/Positive
- 0 Positive confirmation
- 1 Negative confirmation

EMP — Empty Transmit Queue
- 0 If CFD=1 and NPV=1 in the next FD control word then the TBC sets this queue to not empty
- 1 If NPV=0 in the next FD control word then the TBC sets this queue to empty

FTL — Frame Too Long
- 0 No meaning
- 1 The frame is too long. The actual data length is greater than the frame data length from the FD which is set by the host. The TBC transmitted the data contained in the first data buffer or the frame data length set by the host, whichever is the shortest, followed by an abort sequence.

RES — Reserved

UR — Underrun
- 0 No underrun occurred
- 1 Underrun occurred

NR — No Response
- 0 No meaning
- 1 If N/P=1 then no response was received to an RWR frame. If N/P=0 then response was received on one of the following retries.

UF — Unexpected Frame
- 0 No meaning
- 1 Unexpected frame received during RWR

RT7-RT0 - Number of Tries
Number of tries, except if the MAX RETRY LIMIT was reached then it is the number of retries. Refer to **4.4 REQUEST WITH RESPONSE (RWR) TRANSMISSION** and **4.5 RECEPTION OF RWR FRAMES AND TRANSMISSION OF RESPONSE** for further details.

When used in a RECEIVE queue, the frame descriptor confirmation/indication word has the following format and is updated by the TBC:

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
NRV	*	*	*	*	*	*	*	*	*	*	*	*	*	*	*

*Don't Care

NRV — Next Receive Pointer Valid
- 0 The pointer to the next FD in this receive queue is not valid so this is the last frame in this receive queue.
- 1 The pointer to the next FD in this receive queue is valid.

4.1.1.2 RECEIVE STATUS WORD. The receive status word is updated by the TBC in the receive case. The host must clear this word in the frame descriptor before placing the frame descriptor into the free frame descriptor pool. A one in the corresponding bit of the receive status error mask located in the private area will cause the frame to be accepted according to the following criteria:

- If a CRC error occurred the frame is stored in its entirety. The frame data length (offset 10 in FD) reflects the actual data length including the CRC length if the RCDS (RX CRC to data storage mode) is enabled.
- If an E bit error was detected, the frame is stored in its entirety and the frame data length is updated to reflect the number of bytes the TBC heard.
- If a FIFO overflow occurred (i.e., overrun error), the data is copied to the data buffer(s) up to the point where the overrun error occurred. The frame data length reflects the number of bytes copied to the data buffer(s).
- If noise was detected, the data is copied to the data buffer(s) up to the point where the noise occurred. The frame data length reflects the number of bytes copied to the data buffer(s).
- If a frame >8K was detected, the data is copied to the data buffer(s) in its entirety. The frame data length reflects the number of bytes copied to the data buffer(s).
- If not enough buffers are available in the free buffer pool, the TBC receives the frame's header information as long as free FDs are available. The correct frame length is stored in the FD.

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
CRCE	EBIT	FOVF	NOISE	FTL	NOBUF	*	*	*	*	*	*	*	*	*	*

*Reserved

CRCE — CRC error occurred in the frame received
EBIT — The E bit in the end delimiter is set
FOVF — FIFO overflow
NOISE — Noise
FTL — Frame too long (the received frame is longer than 8K bytes)
NOBUF — Not enough buffers in buffer pool

4.1.1.3 CONTROL FOR NEXT FRAME DESCRIPTOR POINTER. This word is updated by the host and its format is shown below:

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
NPV	W_FD	*	*	*	*	*	*	*	*	*	*	*	*	*	*

*Reserved

NPV — Next Pointer Valid
Valid in TX and RX Cases
0 This is the last valid FD in this queue. The TBC will set the FD pool empty bit in interrupt status word 0 upon using this FD for receiving.
1 The next FD in this queue is valid.

W_FD — Warning Frame Descriptor Pool Low
Valid in free pool case. This bit should be used only while in a free frame descriptor pool.
0 During transmission, the W_FD bit must be set to zero. Note that in the last FD the free pool (NPV=0) W_FD must be a zero.
1 FD pool is almost empty, the FD pool low bit in the interrupt status word 0 will be set upon the TBC's detection of this condition.

4.1.1.4 NEXT FRAME DESCRIPTOR POINTER. This 32-bit address points to the next frame descriptor in the queue. This pointer must be initialized by the host if the next FD is valid in both the transmit and free pool cases.

4.1.1.5 FIRST BUFFER DESCRIPTOR POINTER. This 32-bit address is used by the TBC to address the first buffer descriptor. When in a transmit queue, this pointer must be initialized by the host. If this FD resides in the free frame descriptor pool, the TBC will update this pointer as it gets a BD from the free buffer descriptor pool upon receiving a frame.

4.1.1.6 FRAME DATA LENGTH. During transmit, frame data length is updated by the host; during receive, it is updated by the TBC. Normally, the frame data length is the length of the data unit of the frame. The only exception is if RCDS (RX CRC to data storage mode) is enabled, in that case the CRC is included in frame data length.

4.1.1.7 IMMEDIATE RESPONSE FRAME DESCRIPTOR POINTER. This word contains the 32-bit address of the immediate response frame descriptor. This pointer is updated by the TBC. This is used after a station has transmitted a request with response frame and receives a valid response. Then the IR frame descriptor pointer of the RWR frame points to the frame descriptor of the response, thus linking the RWR frame with its response.

4.1.1.8 FRAME CONTROL. The frame control field determines the category of frame that is being sent or received. In the transmit case, the host is responsible for updating this field, while in the receive case the TBC is. Three frame categories are currently defined by IEEE 802.4: MAC control and LLC data. Note that the host is forbidden from setting this byte to a MAC control value in the TX case. See **APPENDIX B FRAME FORMATS AND ADDRESSING** for more details.

4.1.1.9 MAC DESTINATION ADDRESS. The MAC destination address can be either two or six bytes. The address length is determined by the host at initialization time by setting appropriately the address length bit which resides in the initialization table at offset 7A. This address is updated by the host in the transmit case, while in the receive case the TBC is responsible for doing so. Note that this address is stored following the IEEE convention, see **APPENDIX B FRAME FORMATS AND ADDRESSING** for more details.

4.1.1.10 MAC SOURCE ADDRESS. The MAC source address can be either two or six bytes in length. The address length is determined by the host at initialization time by setting appropriately the address length bit which resides in the initialization table at offset 7A. This address is updated by the host in the transmit case, while in the receive case the TBC is responsible for doing so. Note that this address is stored following the IEEE convention, see **APPENDIX B FRAME FORMATS AND ADDRESSING** for more details.

4.1.2 Buffer Descriptors

Buffer descriptors (BD) contain buffer parameters, buffer status, and pointers. Buffer descriptors are chained together linking data buffers which contain frame data. The format of a buffer descriptor is shown in Figure 4-4. Note that a buffer descriptor must start at an even address.

DISPLACEMENT (IN BYTES)	DESCRIPTION OF FIELD
0	RESERVED
2	DATA BUFFER POINTER — HIGH
4	DATA BUFFER POINTER — LOW
6	BD CONTROL AND OFFSET
8	BUFFER LENGTH
0A	RECEIVE INDICATION WORD
0C	RESERVED
0E	NEXT BUFFER DESCRIPTOR POINTER — HIGH
10	NEXT BUFFER DESCRIPTOR POINTER — LOW

Figure 4-4. Buffer Descriptor Format

4.1.2.1 DATA BUFFER POINTER. These two words contain the 32-bit pointer to the data buffer. The data buffer pointer links the buffer descriptor to its associated data buffer. Each buffer descriptor has only one associated data buffer. The buffer descriptor points to its associated data buffer whether the buffer descriptor is being used for a TX frame, an RX frame, or is still in the free buffer descriptor pool. This pointer must be initialized by the host.

4.1.2.2 BUFFER DESCRIPTOR CONTROL AND OFFSET. This word contains the control and offset information for a buffer descriptor and must be initialized by the host. The offset is useful for appending upper layer software information to the data buffer. The offset tells the TBC how far from the beginning of the data buffer actual data begins. This is useful for passing data through the ISO layers. As the message is passed down through the software layers, address and control information may be appended to the front of the data and the offset may be changed for the next ISO layer so that the data does not have to be rewritten. Note that bit F of this word has a different meaning for the TX queue and the free buffer descriptor pool.

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
LABD	OF14	OF13	OF12	OF11	OF10	OF9	OF8	OF7	OF6	OF54	OF4	OF3	OF2	OF1	OF0

LABD — Last Buffer Descriptor, Transmit Queue Case (TBC will use last BD)

0 This BD is not the last one (next buffer descriptor pointer is valid)

1 This is the last linked buffer in this frame

LABD — Last Buffer Descriptor, Free BD Pool Case (TBC will not use last BD while receiving)

0 This buffer descriptor is not the last one

1 This is the last linked buffer in the pool queue and this buffer descriptor is not valid. The TBC will set the BD pool empty bit in interrupt status word 0 upon detection of this condition.

OF14-OF0 — Offset Length in Bytes from the Head of the Buffer where Actual Data Begins

In free BD pool case, buffer pointer plus offset must be even. In transmit queue case, if this is the first BD of the frame then buffer pointer plus offset may be even or odd. In transmit queue case, if this is NOT the first BD of the frame then buffer pointer plus offset must be even. Also in transmit queue case, if a frame has to be divided into two or more buffer descriptors and the first data buffer only holds one byte of data then the BD pointer plus offset must be even.

4.1.2.3 BUFFER LENGTH. This word must be initialized by the host in all cases.

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
W_BD	BL14	BL13	BL12	BL11	BL10	BL9	BL8	BL7	BL6	BL5	BL4	BL3	BL2	BL1	BL0

W_BD — Warning for Buffer Descriptor Pool

0 In the free pool case indicates no warning. In the transmit case this bit must be zero.

1 In the free pool case, indicates the free BD pool is low. The BD pool low bit in interrupt status word 0 will be set upon the TBC's detection of this warning.

BL14-BL0 — Buffer Length

In both the free pool and transmit cases, this value represents the number of data bytes in the data buffer associated with this BD.

4

4.1.2.4 RECEIVE INDICATION WORD. This word is updated by the TBC and only has meaning in the receive case. The host must clear this indication word before adding this BD to the free BD pool.

F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0
RLBD	OL14	OL13	OL12	OL11	OL10	OL9	OL8	OL7	OL6	OL5	OL4	OL3	OL2	OL1	OL0

RLBD — Receive Last Buffer Descriptor

0 Not the last linked buffer in this receive frame

1 This is the last linked buffer in this receive frame

OL14-OL0 — Offset Length

Number of unused data bytes in this buffer

4.1.2.5 NEXT BUFFER DESCRIPTOR POINTER. This 32-bit address points to the next buffer descriptor in the queue. This pointer must be initialized by the host if the next BD is valid in both the transmit and free pool cases.

4.1.3 Data Buffers

A data buffer is a continuous block of memory reserved for message data. Different buffers do not need to be contiguous. Data buffer size can vary from zero bytes to 32K−1, bytes. Buffers are concatenated via buffer descriptors for messages requiring multiple data buffers. The maximum message length which the TBC will store in data buffers is 64K bytes provided the RX status error mask located in the private area is set to accept such frames. However, since the IEEE 802.4 maximum specified message length is 8K bytes as supported by the TBC's ACM, noise will be reported in the interrupt status words along with storing of such frames. There is no equivalent error conditions reported when transmitting a frame of length greater than 8K.

4.2 TRANSMISSION OF A FRAME

The following paragraphs present the actions required from the host to transmit a frame, and the feedback from the TBC upon completion of frame transmission.

4.2.1 Initialization Performed by Host

In order to transmit a frame, the host must prepare the frame descriptors and buffer descriptors as required for the frame. The following data structures must be initialized by the host:

1. A frame descriptor must be created for each frame as shown below:
   - Zero the confirmation/indication word (offset 0)
   - Zero the receive status word (offset 2)
   - Initialize the control word for next FD pointer (offset 4) with NPV (next pointer valid bit) set to one or zero depending on whether or not another frame has to be transmitted. Also, W_FD (warning FD pool low bit) must be set to zero.
   - Initialize the next frame descriptor pointer (offset 6) if another frame has to be transmitted, i.e., NPV = 1.
   - Initialize the pointer for the first buffer descriptor (offset C)
   - Update the frame data length (offset 10) per the actual frame length to be transmitted. Note that the TBC does not check for this value to be <8K bytes. The TBC can transmit frames up to 32K − 1 bytes.
   - Update the frame control (offset 1B). Note that the TBC does not check for the validity of this value.
   - Initialize the MAC destination and source address. Note that the TBC does not check for the validity of these values.

2. Buffer descriptor(s) must be created, the host has to decide how many are required for each frame:
   - Initialize the data buffer pointer (offset 2).
   - Set LABD (last buffer descriptor bit), located in the control and offset word (offset 6), to one or zero depending on whether or not this is the last linked buffer for this frame. Also, the offset in bytes from the head of the data buffer to where actual data begins must be entered.
   - Load the data buffer length (offset 8) in bytes in bits 0 through 14. Bit 15 which is the warning for buffer descriptor pool low must be set to zero.
   - Initialize the next buffer descriptor pointer (offset 0E) if the next BD is valid.

3. The data must be loaded in data buffer(s) as necessary. Note, that only one data buffer may be associated with a buffer descriptor.

4.2.2 Management of Transmission Queues

The TBC gets frames for transmission from four queues, one for each priority class. The following is a description of the possible queue conditions.

EMPTY QUEUE

A transmission queue which the host is certain is empty, that is the queue is either uninitialized or the last frame was confirmed. A queue is not empty if for all FDs in the queue, either CFD is zero or CFD is one and NPV is one.

ACTIVE QUEUE

A transmission queue for which the host is certain that the TBC did not read the control word of its last frame's FD. That is, no confirmation was given by the TBC in the frame preceding the final frame in the queue.

NOT SURE QUEUE

A transmission queue for which the host cannot determine which of the two previous categories the queue belongs to. This occurs when confirmation was given to the frame preceding the final frame, but not the last frame in the queue, so the user is not sure whether the last frame in the queue will be determined to be empty before the next frame is linked onto it.

4.2.3 Examples of TBC Transmission Queues

NPV — Next Pointer Valid

This bit resides in the control word for the next FD.

0 Next pointer is not valid
1 Next pointer is valid

The NPV bit is set and cleared by the host.

CFD — Confirm Frame Descriptor

This bit resides in the confirmation/indication word of the FD

0 TBC did not complete this frame's transmission
1 TBC completed this frame's transmission

EMP — Empty

This bit resides in the confirmation/indication word of the FD

0 This queue is not empty
1 This queue is empty

EMP reflects the value of NPV as read by the TBC. EMP is not valid if CFD = 0 because the host clears it when preparing the frame.

The CFD and EMP bits are set and cleared by the TBC in a single memory access after sampling NPV.

Figure 4-5 is an illustration of each of the status of queues as designated in **4.2.2 Management of Transmission Queues.** Each block represents a frame and the arrows indicate the order of transmission.

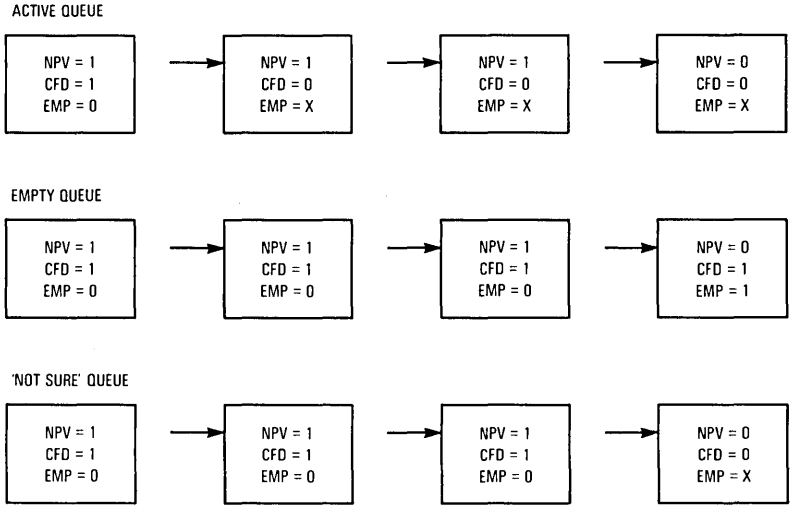


Figure 4-5. Examples of Queues Status

4.2.4 Adding a Frame to a Transmission Queue

To cause the TBC to transmit a frame, four cases have to be examined.

In the first case, let's assume that no frames have been previously sent; i.e., the host is ready to send the first frame after initialization. The head of queue pointer for the desired TX access class which resides in the private area must be initialized to point to the first frame descriptor and the queue's status must be updated. This initialization is accomplished using the START command (see **3.4.3 START Command**). The START command must be issued to start transmission for each queue.

In the second case, frames could have been previously sent by the TBC but the last frame has already been confirmed by the TBC. In this case, as in the previous case, this amounts to adding a frame to an empty queue (see Figure 4-5); therefore, the START command must be issued.

In the third case, the queue to which the frame belongs is active; that is, the next to last frame has not been confirmed. This is equivalent to adding a frame to an active queue as shown in Figure 4-5. To cause the TBC to transmit the newly added frame, the host has to update the pointer to the new frame in the next FD pointer field and set the NPV bit in the last FD of this active queue.

Finally, in the fourth case a frame could be added to a 'not sure' queue. This is accomplished by adding a frame as if to an active queue as above, then waiting for the CFD bit on the previous FD to be updated. The EMP bit is then checked to see if the queue was an active queue or an empty queue. If it was active, then nothing more needs to be done. If it was empty, then a START command must be issued to the TBC as in adding a frame to an empty queue. A 'not sure' queue that turns out to be an empty queue can be detected if there is a confirmed frame with the EMP bit in the confirmation word set and the NPV bit in the control word also set. This event can be detected by an interrupt routine which could be run upon servicing a TBC interrupt triggered by the perceived end of the queue (TXQE, bit 3 of interrupt status word 0).

The host should keep two pointers for each transmit access class queue: one for the head of the queue where the host checks for TBC confirmation of sending frames, and one to the end of the queue where the host links new frames. The TBC, on the other hand, keeps one pointer to the head of each transmission queue located in the private area so it knows where to start transmitting from.

4.2.5 TBC'S Actions Upon Transmission

Upon completion of frame transmission, the TBC will update the confirmation word located in the FD as follows:

- The CFD bit is set.
- The N/P bit is set if the transmission failed and cleared if the transmission succeeded.
- The EMP bit is set if this was the last frame in this transmit queue.
- If the actual frame length, calculated from the sum of the data in the frame's data buffers, is greater than the frame length (offset 10 in FD), then the negative confirmation bit will be set along with the FTL bit.
- The UR bit (underrun) wiil be set if the TBC attempted to transmit from an empty FIFO. The underrun bit in interrupt status 0 will also be set and an interrupt will be generated if enabled. The frame will be retransmitted if the non-RWR max retry limit parameter located in the private area has not been exceeded. The non-RWR max retry limit is not an IEEE 802.4 parameter, its maximum value is 255 and a retry will be performed only if an underrun has occurred.

Lastly, the TBC will set the TXDF bit in interrupt status word 0 (transmitted data frame), and generate an interrupt if enabled.

Once the host has checked that the frame has been processed (the CFD bit is set), it clears the confirmation indication word and either links this FD to the end of the free frame descriptors pool, uses it in the next transmit frame, or frees up memory space.

4.3 RECEPTION OF FRAMES

Every frame detected on the token bus medium has its destination address field checked by each station to see if that station should receive it. The following paragraphs describe the setting up which is required by the host to allow the TBC to store a frame, as well as the mechanism used by the TBC to do so.

4.3.1 Initialization Performed by Host

In order to allow the TBC to store the received frames, the host must set up free frame descriptor and buffer descriptor pools as shown in the example of Figure 4-6.

To enable the TBC to store incoming frames, the host must prepare the following data structures:

1. The free frame descriptor pool pointer located in the private area must be updated at initialization time or through the SET TWO WORDS command. This pointer must point to a valid free FD. Refilling of the FD pool is accomplished by writing the pointer to a new FD in the last FD.

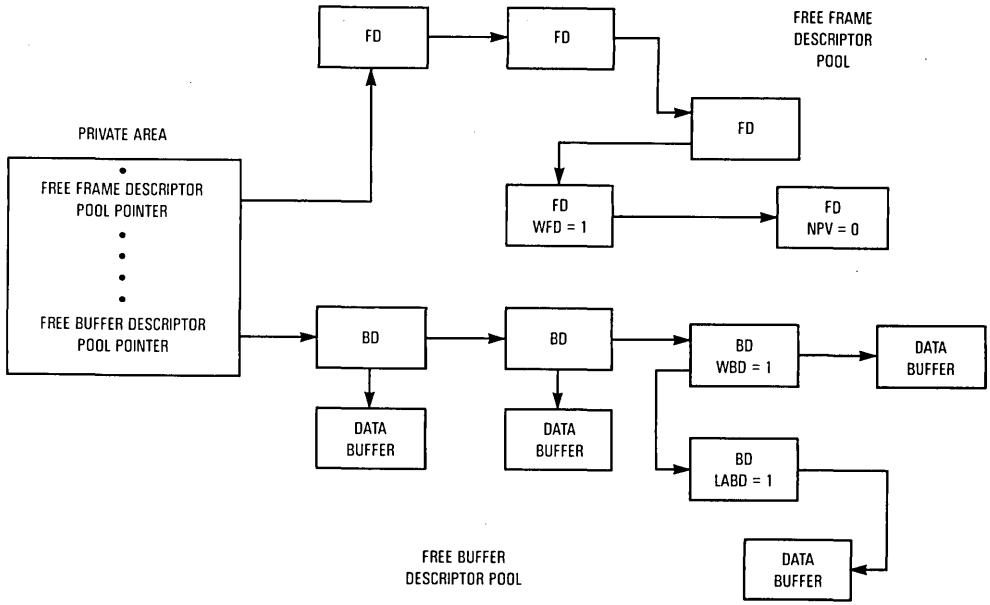


Figure 4-6. Free Frame Descriptor and Buffer Descriptor Pools

2. Each FD in the free pool must be initialized as follows:
   - Zero the confirmation/indication word (offset 0).
   - Zero the receive status word (offset 2).
   - Initialize the control word for next FD pointer (offset 4) with NPV (next pointer valid bit) set to one or to zero depending on whether or not this is the last valid FD. Also, W_FD (warning FD pool low bit) may be set to one or zero to enable the host to be notified when the FD pool is low whenever applicable. W_FD must not be set in the last FD.
   - Initialize the next frame descriptor pointer (offset 6) if the next FD is valid; i.e., NPV = 1.
3. The free buffer descriptor pool pointer located in the private area must be updated at initialization time or through the SET TWO WORDS command. This pointer must point to a valid free BD. Refilling of the BD pool is accomplished by writing the pointer to a new BD in the last BD.
4. Each BD in the free pool must be initialized as follows:
   - Initialize the data buffer pointer (offset 2).
   - Set LABD (last buffer descriptor bit) to one or zero in the control and offset word (offset 6), depending on whether or not this is the last buffer available in the free pool. The TBC will not use the last buffer. Also, the offset in bytes from the head of the data buffer to where actual data should be stored must be initialized.
   - Load the data buffer length (offset 8) in bytes in bits 0 through 14. Bit 15 (W_BD) may be set to one to indicate that the free BD pool is low. W_BD must never be set to one in the last BD.
   - Clear the receive indication word (offset 0A)
   - Initialize the next buffer descriptor pointer (offset 0E) if the next BD is valid, i.e., LABD = 0.
5. Memory space must be reserved for the data buffers pointed to by the BD's.

4.3.2 Reception Queues and Free Pools

Four queues are maintained for received frames — one for each priority class. Note that these queues may not be disabled by the host. At initialization time, the host must set the RX EOQ pointer for each access class. These pointers may be modified thereafter by using the SET TWO WORDS command. The TBC will update these pointers upon completion of frame reception.

The host is responsible for creating and adding frame descriptors to the free pool. When the last FD in the free pool has been used by the TBC (i.e., it has been linked to the appropriate receive queue), the TBC keeps an internal pointer to this FD as its link to the free pool. Therefore, the host must not remove the last FD from a receive queue. The host can add new FDs to the free FD pool by writing the pointer to a new FD in the last FD in the pool and setting its NPV bit. The TBC polls this last NPV bit on every received frame to see if the host has added FDs to an empty pool.

The host is also responsible for creating and adding buffer descriptors to the free BD pool. The TBC will not use the last buffer descriptor in the free pool. To add buffer descriptors to the pool, the host writes the pointer to the BD of the new buffer in the last buffers BD, and clears its LABD bit. The TBC polls this LABD bit on every received frame to see if the host has added buffers to an empty BD pool.

Refer to Figure 4-7 for an illustration of the initialization of data structures by the host needed to receive frames. This example only shows the linking for one access class because the mechanism

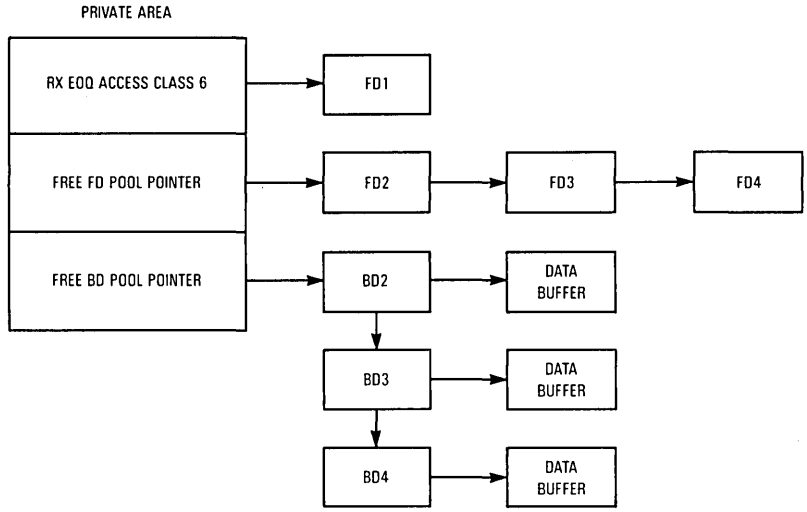


Figure 4-7. Initalization of a Reception Queue

for the other access classes is the same. Each receive queue must consist of at least one dummy FD (FD1). The dummy FD is used to start the linking procedure and contains the pointer to the next FD when there is one as shown in Figure 4-8. Therefore, the host must not remove the last FD, which serves as the dummy FD, from the queue. In Figure 4-8, the TBC has received one frame and has stored it using FD2 and two BDs (BD2 and BD3).

4.3.3 TBC's Actions Upon Reception

As the TBC accepts frames addressed to it, it takes frame descriptors and buffer descriptors from the beginning of the free frame descriptor and free buffer descriptor pools.

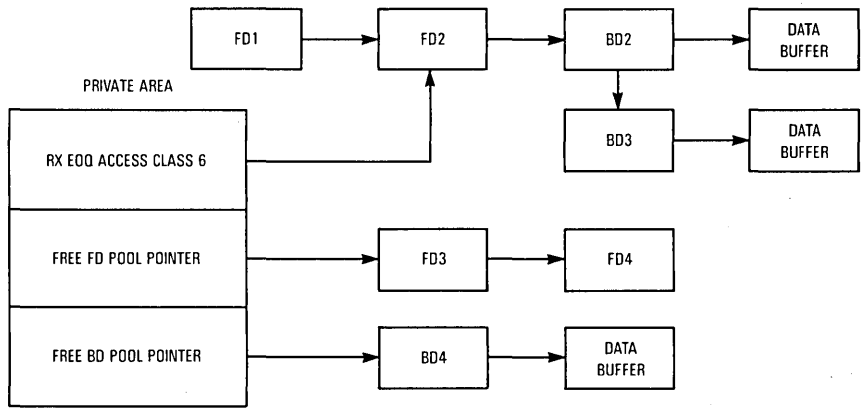


Figure 4-8. Reception Queue After Receiving One Frame

Upon frame reception, the TBC will perform the following steps:

1. For each FD which is pulled off the free pool the TBC will:
   - Update the indication word (offset 0) bit 15 (NRV). If this is the last frame in this RX queue, then NRV=0 else NRV=1.
   - Update the receive status word (offset 2) to reflect possible errors found while receiving this frame. The TBC will accept the frame in which errors were found according to the RX status error mask located in the private area (see **2.1.14 RX Frame Status Error Mask**).
   - Check the control word for next FD pointer (offset 4). If the NPV bit is cleared, the TBC will set the FDPE bit in interrupt status word 0, and this FD will be used by the TBC. Also, the W_FD bit will be checked and the FDPL bit in interrupt status word 0 will be set if required.
   - Update the first BD pointer (offset C) as it fetches a BD from the free pool if the first BD fetched has LABD=0.
   - Load the frame data length (offset 10).
   - Load the frame control (offset 1B), the destination and source address.
2. For each BD which is fetched from the free pool the TBC will:
   - Check bit 15 of the BD control and offset word (offset 6). If LABD=1, then the TBC will NOT use this last BD. The buffer descriptor pool empty bit in interrupt status word 0 will be set. If the RX status error mask permits it, the TBC will receive the frame's header information (i.e., the information stored in the FD) provided a free FD is still available.
   - Check bit 15 of the buffer length (offset 8). If W_BD=1, then the TBC will set the BD pool low bit in interrupt status word 0.
   - Update the receive indication word (offset A) to inform the host if this is the last BD for this frame and the number of unused data bytes in this buffer.
3. The TBC will load the data from the frame into the data buffer(s) associated to the BD(s) which was (were) pulled from the free pool.
4. The TBC will check the private area to find the last FD residing in the appropriate receive queue. It will change this last FD's next receive valid (NRV) bit to a one indicating that there is at least one more FD in the queue. The TBC will then link the received FD to the last FD in the queue, and update the RX EOQ pointer for that access class in the private area.
5. Lastly, the TBC will set the RXDF bit (received data frame) located in interrupt status word 0 and generate an interrupt if enabled.

As the host reads received frames, it must keep at least one FD linked to the queue for each access class to know where to start reading. Before adding FDs to the FD pool, the host must clear the confirmation/indication word in the frame descriptor as well as the status indication word. The host can then free up memory space or add the FD or BD back into the linked list.

4.4 REQUEST WITH RESPONSE (RWR) TRANSMISSION

The mechanism used by the host to prepare an RWR frame for transmission is the same as for any other frame with one exception: the frame control field for that frame (offset 1B of FD) must have RWR encoded in it. When the TBC reaches this frame in the transmit queue, it transmits the frame and waits three slot times for a response as defined in IEEE 802.4. If no valid response arrives, the TBC will try to transmit the frame again, up to the RWR max retry limit which is specified by the user in offset 6E of the private area. IEEE specifies this value to be programmable and to have a maximum value of seven, however the TBC allows a maximum value of 255 for

this parameter. Note that an underrun also counts as a retry. If a valid response arrives, the TBC goes through the following steps:

- Loads the frame into the appropriate priority receive queue.
- Links the transmitted RWR frame to the response frame by placing the pointer to the response's FD in the transmitted RWR frame's immediate response FD pointer in the transmitted frame's FD.
- Gives confirmation in the RWR frame's FD.
- Sets TXRWR bit (transmitted request with response) in interrupt status word 0 and interrupts the host if enabled by the interrupt status mask.

If no response is received after the RWR retry limit is reached, the TBC gives negative confirmation on this frame in the RWR's FD's confirmation/indication word, updates the interrupt status word to indicate TXRWR, and interrupts the host if enabled.

A valid response is defined as an LLC data frame with a MAC action field set to response which is received by the requester within the RWR max retry limit and before any control frames or other data frames are received.

If the TBC hears an unexpected frame while waiting for a response, the TBC will:

- Store that frame into the appropriate queue.
- Give a negative confirmation as well as set the unexpected frame received during RWR bit in the RWR's FD.
- Set the TXRWR bit in interrupt status word 0 as well as the unexpected frame 6 bit in interrupt status word 1.

4.5 RECEPTION OF RWR FRAMES AND TRANSMISSION OF RESPONSE

Upon receiving an RWR frame addressed to this station, the TBC will store it into the appropriate receive queue and perform the following steps:

- The source address of the received frame is stored into the destination address field of the FD pointed to by the response destination address pointer. This pointer resides in the initialization table at offset 84 and must have been previously initialized by the host to point to a valid FD.
- If the frame is not a retry, then the TBC loads the pointer to the received RWR frame's FD into the RWR pointer field located in the initialization table at offset 8C. The received RWR frame bit in interrupt status word 0 will then be set and an interrupt will be generated if enabled.

A retry is defined as an RWR frame addressed to this TBC that arrives immediately after another RWR frame which was addressed to it. If another valid frame such as a protocol frame, a non-RWR data frame, or a data frame not addressed to this station arrives in between two RWR frames, then the second RWR frame is considered to be a new frame and not a retry.

The response is sent by the TBC according to the steps defined below:

- If the TBC is in predefined response mode, set by the host through the SET MODE 1 command, then the pointer stored by the host in the response pointer field in the initialization table (offset 88) is considered valid, and the response is transmitted immediately after the token bus goes to silence. The TXRDF bit (transmitted response data frame) in interrupt status word 0 is also set and an interrupt is generated if enabled.

- If the TBC is not in predefined response mode, then the TBC must wait for the command RESPONSE READY from the host before sending out the response. Before issuing the RESPONSE READY command, the host must prepare a response and update the response pointer field located in the initialization table (offset 88) if necessary. If this command is issued within two slot times, then the TBC will transmit the response prepared by the host.

  However, if the host issues the command later than two slot times but before the TBC receives the next RWR retry frame, the TBC will still transmit the previously prepared response as a response to the current retry of the RWR frame. The TXRDF bit (transmitted response data frame) in interrupt status word 0 is also set and an interrupt is generated if enabled.

Note that if the host still owes a previous response to the TBC, the TBC will neither update the RWR pointer field nor change the status of RXRWR in interrupt status word 0.

The following four paragraphs describe the relationship between two of the pointers residing in the initialization table at offset 84 and 88 which are used to implement the RWR mechanism.

Case 1:

The TBC is in predefined response mode and the DA of the response frame must be identical to the SA of the received RWR frame.

- The response destination address pointer (offset 84) should have been previously set by the host to the same value as the response pointer (offset 88).
- The TBC will set DA equal to SA of received frame in the FD pointed to by the response destination address pointer (offset 84).

Case 2:

The TBC is in predefined response mode and the DA of the response frame must be different from the SA of the received RWR frame.

- The response destination address pointer (offset 84) should have been previously set by the host to point to a dummy FD.
- The response pointer (offset 88) should have been previously set by the host to point to the response frame's FD where the host has set DA as required.

Case 3:

The TBC is in non-predefined mode and the DA of the response frame must be identical to the SA of the received RWR frame.

- The response destination address pointer (offset 84) should have been previously set by the host to the same value as the response pointer (offset 88).
- The response pointer (offset 88) must be a constant, i.e., always point to the same FD.
- The TBC will set DA equal to SA of received frame in the FD pointed to by the response destination address pointer (offset 84).

Case 4:

The TBC is in non-predefined response mode and the DA of the response frame must be different from the SA of the received RWR frame.

- The response destination address pointer (offset 84) should be set by the host to point to a dummy FD.
- The response pointer (offset 88) is set by the host to point to the response frame's FD where the host has set DA as required upon getting the RXRWR interrupt (this bit could also be polled).

NOTE

Cases 2 and 4 describe operations which are inappropriate for IEEE 802.4.

WARNING

The TBC may send a response to a previous RWR frame as a response to a new RWR frame in the following cases:

- If a second, new RWR frame arrives immediately after a RWR frame whose response was not ready within two slot times from its arrival (this new frame is considered to be a retry).
- If the response from the host is issued to the TBC at the same time as a second RWR frame arrives, even if a non-RWR frame addressed to this station arrived previously. In this case, the RESPONSE READY command is executed only after complete reception of the RWR frame. Thus, the response is considered a valid response to the second request and will be transmitted. This scenario also can take place if the response was ready to be transmitted but, due to high load on the bus, the TBC was not able to execute the RESPONSE READY command prior to receiving the second RWR frame.

Note that in these cases, the host did not respond in time to the last retry of a RWR frame, while according to the 802.4 standard, the host should respond in time to the first one. Thus, these situations should not occur.

SECTION 5
SIGNALS

This section contains a description of the input and output signals for the MC68824 Token Bus Controller.

NOTES

The terms **assertion** and **negation** will be used extensively. This is done to avoid confusion when dealing with a mixture of "active low" and "active high" signals. The term "assert" or "assertion" is used to indicate that a signal is active or true, independent of whether that level is represented by a high or low voltage. The term "negate" or "negation" is used to indicate that a signal is inactive or false.

The TBC has two modes of bus operation, the master mode and the slave mode. In the master mode, the TBC has control of the address and data bus and is performing memory I/O. In the slave mode, the TBC is acting as a peripheral and is being accessed by the host. Refer to **6.1 HOST PROCESSOR OPERATION MODE** and **6.2 DMA OPERATION** for further details.

Input and output signals are functionally organized into the groups shown in Figure 5-1. Each of these groups is discussed in the following paragraphs.

5.1 ADDRESS BUS (A1-A31)

This is a 32-bit (when combined with $\overline{\text{UDS}}$/A0 signal), unidirectional (with the exception of A1 and A2 which are bidirectional), three-state bus capable of addressing up to 4 gigabytes of memory. A1 and A2 are used to address internal registers in slave mode.

5.2 DATA BUS (D0-D15)

The TBC has a 16-bit, bidirectional, three-state bus for a general purpose data path. It can transmit and receive data in either word or byte lengths. In 8-bit mode, data transfer is restricted to D0-D7 pins.

5.3 BUS CONTROL

The following paragraphs describe the bus control signals.

5.3.1 Function Codes (FC0-FC3)

The TBC function code pins are three-state outputs which are asserted during a DMA bus cycle. These function codes may be used to divide memory into separate address spaces. The TBC does not check for any particular 4-bit values. The host processor memory management unit can define

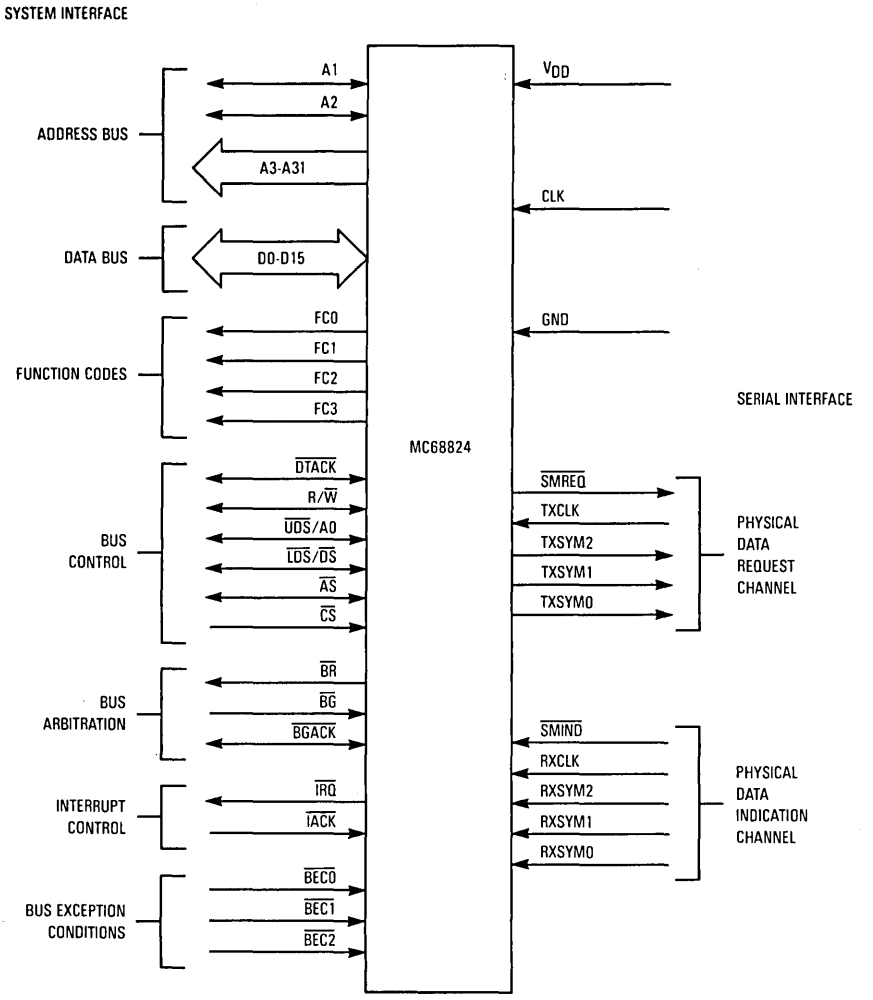


Figure 5-1. MC68824 Signals

these encodings as desired. The user is not required to implement function codes. If the function code pins are not connected, the user can ignore SET FUNCTION CODE commands (see **3.5.3 Set Value**).

5.3.2 Bus Exception Conditions ($\overline{BEC0}$-$\overline{BEC2}$)

These input lines provide an encoded signal that indicates an abnormal bus condition such as a bus error or reset. The three-bit bus exception control codes allow for eight different bus termination conditions. $\overline{BEC0}$ is the least significant bit and $\overline{BEC2}$ is the most significant bit. Table 5-1 shows the eight conditions.

Table 5-1. Bus Exception Conditions

$\overline{BEC2}$	$\overline{BEC1}$	$\overline{BEC0}$	Condition
H	H	H	No Exception
H	H	L	Halt (Release Bus)
H	L	H	Bus Error
H	L	L	Retry
L	H	H	Relinquish and Retry
L	H	L	Undefined (Reserved)
L	L	H	Undefined (Reserved)
L	L	L	Reset

5.3.3 Data Transfer Acknowledge (DTACK)

DTACK is a bidirectional three-state signal which indicates that an asynchronous bus cycle may be terminated. In the slave mode, this output indicates that the TBC has accepted data from the host processor or placed data onto the bus for the host processor. In the master mode, this input is monitored by the TBC to determine when to terminate a bus cycle. As long as DTACK remains negated, the TBC will insert wait cycles into the bus cycle. When DTACK is asserted, the bus cycle will be terminated.

5.3.4 Read/Write (R/W)

R/W is a bidirectional three-state signal that indicates the direction of the data transfer during a bus cycle. The R/W pin is an input in the slave mode and an output in master mode. Thus, in the slave mode, a high level indicates that a transfer is from the TBC onto the data bus, and a low level indicates that a transfer is from the data bus into the TBC. In the master mode, a high level indicates that a transfer is from the data bus into the TBC, and a low level indicates that a transfer is from the TBC onto the data bus.

5.3.5 Upper Data Strobe (UDS/A0), Lower Data Strobe (LDS/DS)

These bidirectional three-state signals control the flow of data onto the data bus. UDS and LDS are asserted by the TBC when operating in the master mode and by the host processor when operating in slave mode. In the master mode, during any 16-bit bus cycle, UDS is asserted if data is to be transferred over the data lines D8-D15, and LDS is asserted if data is to be transferred over data lines D0-D7. For an 8-bit bus configuration, UDS functions as A0 and LDS functions as data strobe. A0 is thus an extension of the lower address lines to provide the address of a byte in the address map and is valid when A1-A31 are valid. DS is a data strobe used to enable external data buffers and indicates that valid data is on the bus during a write cycle. In the slave mode, UDS and LDS operate in conjunction with A1, A2, and CS as indicated in **6.6 REGISTERS** and **5.5.5 Chip Select (CS)**. Figure 5-2 shows the various signal combinations and when data is valid on the data lines.

5.3.6 Address Strobe (AS)

This bidirectional three-state signal indicates that there is a valid address on the address bus. It is an output when the TBC is in master mode and has control of the bus. AS is an input during

UDS	LDS	R/W	D8-D15	D0-D7
16-BIT TRANSFER				
HIGH	HIGH	X	NO VALID DATA	NO VALID DATA
LOW	LOW	X	VALID DATA 8-15	VALID DATA 0-7
HIGH	LOW	LOW	NO VALID DATA	VALID DATA 0-7
HIGH	LOW	HIGH	X	VALID DATA 0-7
LOW	HIGH	LOW	VALID DATA 8-15	NO VALID DATA
LOW	HIGH	HIGH	VALID DATA 8-15	X
8-BIT TRANSFER				
X	LOW	LOW	NO VALID DATA	VALID DATA 0-7
X	LOW	HIGH	X	VALID DATA 0-7
X	HIGH	X	NO VALID DATA	NO VALID DATA

X — Don't Care

Figure 5-2. Data Strobe Control of Data Bus in Master Mode

bus arbitration cycles, and it is monitored to determine when the TBC can take control of the bus (after the TBC has requested and been granted use of the bus).

5.3.7 Chip Select ($\overline{CS}$)

This input pin is used by the TBC to determine when it has been selected by the host processor. When $\overline{CS}$ is asserted, the address on A1, A2, and the data strobes select the internal TBC register that will be involved in the transfer. $\overline{CS}$ should be generated by qualifying an address decode signal with the address and data strobes. Asserting $\overline{CS}$ when the TBC is bus master will cause an address error to occur.

5.4 BUS ARBITRATION

The signals in this group form a bus arbitration circuit that determines which device is the current bus master.

5

5.4.1 Bus Request ($\overline{BR}$)

This open-drain ouptut pin is wire-ORed with all other devices that could be bus masters and is asserted to request control of the bus.

5.4.2 Bus Grant ($\overline{BG}$)

$\overline{BG}$ is an input that indicates to the TBC that bus control has been granted and that it may assume bus mastership as soon as the current bus cycle is completed. The TBC will not take control of the bus until $\overline{AS}$ and $\overline{BGACK}$ are negated and the $\overline{BEC}$ pins are encoded as normal mode.

5.4.3 Bus Grant Acknowledge ($\overline{BGACK}$)

$\overline{BGACK}$ is a bidirectional three-state signal. When $\overline{BGACK}$ is an asserted input signal to the TBC, it indicates that some other device has become the bus master. This signal will not be asserted as an output to indicate that the TBC is the current bus master until the following conditions are met:

1. Bus request ($\overline{BR}$) is asserted.
2. A bus grant ($\overline{BG}$) has been received.
3. Address strobe ($\overline{AS}$) is inactive, indicating that the current bus cycle has ended.
4. Bus grant acknowledge ($\overline{BGACK}$) is inactive, which indicates that no other device is still claiming bus mastership.
5. $\overline{BEC}$ pins are encoded as normal mode.

5.5 INTERRUPT CONTROL

Two signals are used to perform the request/acknowledge handshake function between the TBC and the host processor.

5.5.1 Interrupt Request ($\overline{IRQ}$)

The interrupt request pin is an open-drain output pin which requests service from the host processor when an interrupting event has occurred.

5.5.2 Interrupt Acknowledge ($\overline{IACK}$)

This input is asserted by the host processor to acknowledge that it has received an interrupt from the TBC. The TBC responds by placing a vector on D0-D7 that is used by the host processor to fetch the address of the proper TBC interrupt handler routine. The TBC does not negate $\overline{IRQ}$ after an $\overline{IACK}$ cycle, but rather after the appropriate bit(s) in the interrupt status registers is (are) cleared by the host processor.

5.6 SERIAL INTERFACE

The serial interface implemented on the TBC complies with the IEEE Draft Standard 802.4G which describes the exposed DTE to DCE interface. This interface provides a standard way to transfer requests for data unit transmission and for indicating data unit reception when in the MAC mode. The interface also serves as a communication channel for passing station management requests, indications, and confirmations between the physical layer and the TBC. Note that since the TBC is a half duplex part, it will not hear its own transmission. The following paragraphs describe the data request and data indication channel while Figure 5-3 illustrates the signals comprising the physical layer interface.

5.6.1 Physical Data Request Channel

Five signals comprise the physical data request channel with four of these being outputs and TXCLK being an input. Three of these signals are multiplexed and have different meanings depending on the mode of operation. The $\overline{SMREQ}$ signal determines whether the physical layer is in station management mode or MAC mode. When programmed for MAC operation, this channel

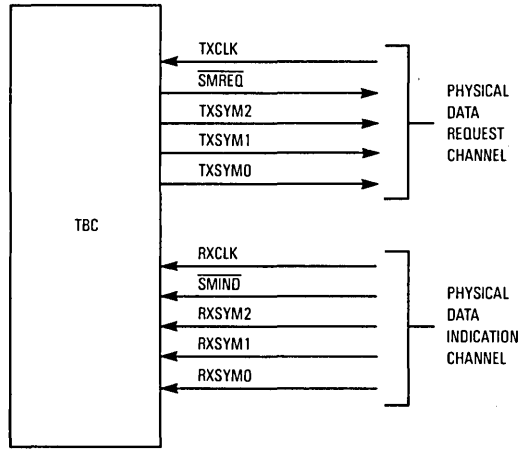


Figure 5-3. Physical Layer Interface

is used by the MAC to provide encoded requests (atomic symbols) for data transmission. In station management mode, this channel is used to pass commands or send serial station management data to the physical layer.

5.6.1.1 STATION MANAGEMENT REQUEST ($\overline{\text{SMREQ}}$). $\overline{\text{SMREQ}}$ is an output used to select the modem mode of operation. $\overline{\text{SMREQ}}$ is called TXSYM3 in the 802.4G Draft Standard. When $\overline{\text{SMREQ}}$ is equal to one, MAC mode is selected. This is considered the normal mode of operation. The MAC sends data to the physical layer encoded as atomic symbols (silence, non-data, pad-idle, data 'one', and data 'zero') on lines TXSYM0, TXSYM1, and TXSYM2 in normal operation. The atomic symbols request data unit transmission, and the physical layer modulates its transmit carrier signal accordingly. When $\overline{\text{SMREQ}}$ is equal to zero, station management mode is selected.

5.6.1.2 TRANSMIT CLOCK (TXCLK). The transmit clock is supplied to the TBC by the physical layer and can be from 10 kHz to 16.67 MHz. TXSYM2, TXSYM1, and TXSYM0 are synchronized to TXCLK.

5

5.6.1.3 TXSYM2, TXSYM1, AND TXSYM0 IN MAC MODE. TXSYM2, TXSYM1, and TXSYM0 are all output signals. When operating in the MAC mode, the request channel symbol encodings for TXSYM2, TXSYM1, and TXSYM0 are those shown in the Figure 5-4. When the TBC does not have any data to transmit it will encode ones on the TXSYM lines, that is silence will be transmitted. The TBC looks for silence on the RXSYM lines to determine if it can transmit. Also note that the TBC does not receive its own transmission being a half duplex part.

SYMBOL	TXSYM2	TXSYM1	TXSYM0
DATA ZERO	0	0	0
DATA ONE	0	0	1
PAD-IDLE	0	1	X
NON-DATA	1	0	X
SILENCE	1	1	X

Where:
DATA ZERO is a binary zero
DATA ONE is a binary one
NON-DATA is a delimiter flag and is always requested in pairs
SILENCE is silence or pseudo-silence
PAD-IDLE is one symbol of preamble/interframe idle

Figure 5-4. Request Channel Encodings for MAC Mode ($\overline{\text{SMREQ}}$=1)

5.6.1.4 TXSYM2, TXSYM1, AND TXSYM0 IN STATION MANAGEMENT MODE. The encoding for the request channel signals TXSYM2, TXSYM1, and TXSYM0 in station management mode is shown in Figure 5-5. By setting $\overline{\text{SMREQ}}$ equal to zero, the request channel enters station management mode. When in station management mode, the request channel can command the physical layer to reset, disable loopback, enable transmitter, idle, or send SM data serially to the physical layer.

The PHYSICAL and END PHYSICAL commands which are described in **3.8 MODEM CONTROL** are the means by which the host sends commands through the TBC to the physical layer.

SYMBOL	TXSYM2	TXSYM1	TXSYM0
RESET	1	1	1
LOOPBACK DISABLE	1	0	1
ENABLE TRANSMITTER	0	1	1
SERIAL SM DATA ZERO OR START BIT	0	0	0
SERIAL SM DATA ONE OR STOP BIT OR IDLE	0	0	1

Figure 5-5. Request Channel Encodings for Station Management Mode ($\overline{SMREQ}=0$)

5.6.2 Physical Data Indication Channel

Five input signals comprise the physical data indication channel. Three of these signals are multiplexed and have different meanings depending on the mode of operation. The $\overline{SMIND}$ signal determines if the channel is in station management mode or MAC mode. When programmed for MAC operation, this channel is used by the physical layer to provide encoded indications of the data unit reception. In station management mode, this channel either carries confirmations to commands passed to the physical layer or reports physical layer conditions.

5.6.2.1 STATION MANAGEMENT INDICATION ($\overline{SMIND}$). $\overline{SMIND}$ is an input to the TBC which indicates whether the physical layer is in MAC mode ($\overline{SMIND}=0$) or station management mode ($\overline{SMIND}=1$). $\overline{SMIND}$ is called RXSYM3 in the 802.4G Draft Standard. When in MAC mode, the physical layer sends encoded indications of data unit reception (silence, non-data, bad signal, data 'one', and data 'zero'). The physical layer enters management mode ($\overline{SMIND}=0$) to send responses to management commands at the request of the TBC, to report an error event at the initiative of the modem, or to report real time data also at the initiative of the modem.

5.6.2.2 RECEIVE CLOCK (RXCLK). The receive clock is supplied to the TBC by the physical layer and can be from 10 kHz to 16.67 MHz. RXSYM0, RXSYM1, RXSYM2, and $\overline{SMIND}$ are synchronized to RXCLK.

5.6.2.3 RXSYM0, RXSYM1, AND RXSYM2 IN MAC MODE. Figure 5-6 shows the encoding of RXSYM0, RXSYM1, and RXSYM2 in MAC mode.

SYMBOL	RXSYM2	RXSYM1	RXSYM0
DATA ZERO	0	0	0
DATA ONE	0	0	1
BAD-SIGNAL	0	1	X
NON-DATA	1	0	X
SILENCE	1	1	X

Where:
DATA ZERO is a binary zero
DATA ONE is a binary one
BAD-SIGNAL indicates reception of a bad signal for one MAC symbol period
NON-DATA is a delimiter flag. In the absence of errors, NON-DATA will always be present in pairs
SILENCE is silence or pseudo-silence for one MAC symbol period

Figure 5-6. Indication Channel Encodings for MAC Mode ($\overline{SMIND}=1$)

5.6.2.4 RXSYM2, RXSYM1, AND RXSYM0 IN STATION MANAGEMENT MODE. When in station management mode, the physical layer sends confirmation to commands from the TBC (reset, loopback disable, enable TX, and serial station management data) using the indication channel. This is accomplished by the encodings on signals RXSYM2, RXSYM1, and RXSYM0 as shown in Figure 5-7. Figure 5-8 illustrates a typical station management sequence.

STATE	RXSYM2	RXSYM1	RXSYM0
ACK (ACKNOWLEDGEMENT)	0	1	*
NACK (NON-ACKNOWLEDGEMENT)	1	0	*
IDLE	0	0	1
PHYSICAL LAYER ERROR	1	1	1

Where * indicates RXSYM0 contains the SM data when responding to a serial data command. A '1' in that position indicates a serial SM data one or stop bit. A '0' in that position indicates a serial SM data zero or start bit.

Figure 5-7. Encodings for Station Management Mode ($\overline{SMIND}=0$)

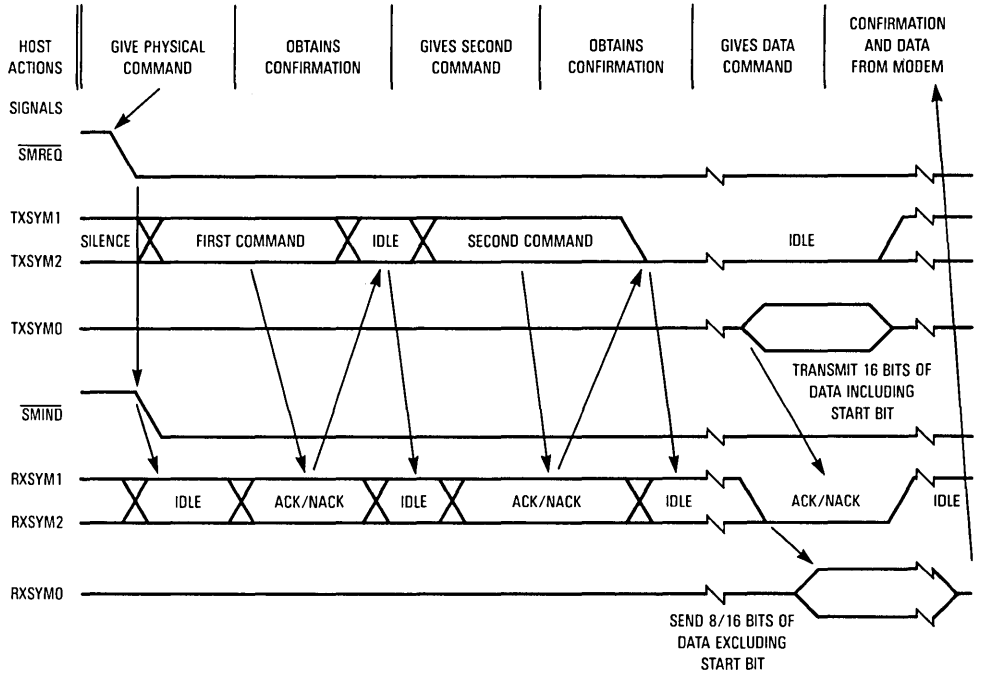


Figure 5-8. Typical Station Management Sequence

The mechanisms by which the TBC obtains acknowledgements from the physical layer are described in **3.8 MODEM CONTROL.** If the TBC receives a physical layer error encoding while in the MAC mode, it will put itself in the offline mode and set the modem error bit in interrupt status word 1 (see **2.2.11 Interrupt Status Words**).

5.7 SIGNAL SUMMARY

Table 5-2 is a summary of all signals discussed in the previous paragraphs.

Table 5-2. Signal Summary

Signal Name	Menemonic	Input/Output	Active State	Driver Type
Address Bus	A1-A2	Input/Output		Three State
Address Bus	A3-A31	Output		Three State
Data Bus	D0-D15	Input/Output		Three State
Function Codes	FC0-FC3	Output		Three State
Bus Exception Codes	$\overline{BEC0}$-$\overline{BEC2}$	Input	Low	
Upper Data Strobe	$\overline{UDS}$/A0	Input/Output	Low/High	Three State[1]
Lower Data Strobe	$\overline{LDS}$/$\overline{DS}$	Input/Output	Low/Low	Three State[1]
Address Strobe	$\overline{AS}$	Input/Output	Low	Three State[1]
Read/Write	R/$\overline{W}$	Input/Output	High/Low	Three State[1]
Chip Select	$\overline{CS}$	Input	Low	
Data Transfer Acknowledge	$\overline{DTACK}$	Input/Output	Low	Three State[1]
Bus Request	$\overline{BR}$	Output	Low	Open Drain[2]
Bus Grant	$\overline{BG}$	Input	Low	
Bus Grant Acknowledge	$\overline{BGACK}$	Input/Output	Low	Three State[1]
Receive Clock	RXCLK	Input		
Indication Channel	$\overline{SMIND}$	Input	Low	
Indication Channel	RXSYM0	Input		
Indication Channel	RXSYM1	Input		
Indication Channel	RXSYM2	Input		
Transmit Clock	TXCLK	Input		
Request Channel	$\overline{SMREQ}$	Input	Low	
Request Channel	TXSYM0	Output		
Request Channel	TXSYM1	Output		
Request Channel	TXSYM2	Output		
Interrupt Request	$\overline{IRQ}$	Output	Low	Open Drain[2]
Interrupt Acknowledge	$\overline{IACK}$	Input	Low	
Clock	CLK	Input		

NOTES:
1. These signals require a pullup resistor to maintain a high voltage when in the high-impedance or negated state. However, when these signals go to the high-impedance or negated state they will first drive the pin high momentarily to reduce the signal rise time.
2. These signals are wire-ORed and require a pullup resistor to maintain a high voltage when not driven.

SECTION 6
BUS OPERATION

The following section describes the bus signal operation during data transfer operations, bus exception conditions, host processor operations, DMA operations, and reset. Functional timing diagrams are included to assist in the definition of signal timing; however, these diagrams are not intended as parametric timing definitions. For detailed relationships refer to **SECTION 8 ELECTRICAL SPECIFICATIONS.**

6.1 HOST PROCESSOR OPERATION MODE

In the host processor operation mode or slave mode, the TBC is a peripheral slave to the bus master. The TBC enters the slave mode when chip select ($\overline{CS}$) or interrupt acknowledge ($\overline{IACK}$) is asserted. During host processor operations, the TBC places data onto the data bus or accepts data from the data bus, respectively, according to the level of the R/$\overline{W}$ pin. This mode of operation is used during TBC initialization to load the configuration information and initial parameters into the TBC. After initialization, the slave mode of operation is used by the host processor to place commands into the TBC command register.

The TBC can operate with a CLK input that is either synchronous or asynchronous with respect to the host processor clock provided the bus requirements are satisfied. The timing diagrams assume that an M68000 Family processor is the host processor with a clock signal identical to the TBC CLK.

6.1.1 Host Processor Read Cycles

During the host processor read cycle, the bus master asserts chip select ($\overline{CS}$), read (R/$\overline{W}$) and lower data strobe ($\overline{LDS}$/$\overline{DS}$). The TBC then places data onto the data bus and asserts data transfer acknowledge ($\overline{DTACK}$) to indicate to the bus master that the data is valid. The semaphore register will always be selected on a read cycle regardless of A1 and A2 encodings because it is the only readable register. When the lower data strobe ($\overline{LDS}$/$\overline{DS}$), or chip select ($\overline{CS}$), is negated by the host, the TBC will three-state the data lines and then negate and three-state $\overline{DTACK}$. This is shown in Figure 6-1. The timing for even and odd byte host reads on a 16-bit data bus or any host read on an 8-bit data bus are identical. The encodings of $\overline{UDS}$/A0 and $\overline{LDS}$/$\overline{DS}$ select the proper byte.

6.1.2 Host Processor Write Cycles

During host processor write cycles, the TBC accepts data from the data bus and asserts $\overline{DTACK}$ to indicate to the bus master that the data has been loaded into the selected register. The only TBC registers that are directly writable by the host processor are the command register, the upper six bits of the interrupt vector register, and the data register.

To begin a host processor write cycle, the bus master asserts $\overline{CS}$ and drives R/$\overline{W}$ low. The TBC responds by decoding the A1, A2, $\overline{UDS}$/A0, and $\overline{LDS}$/$\overline{DS}$ signals. When a valid register (CR, DR,

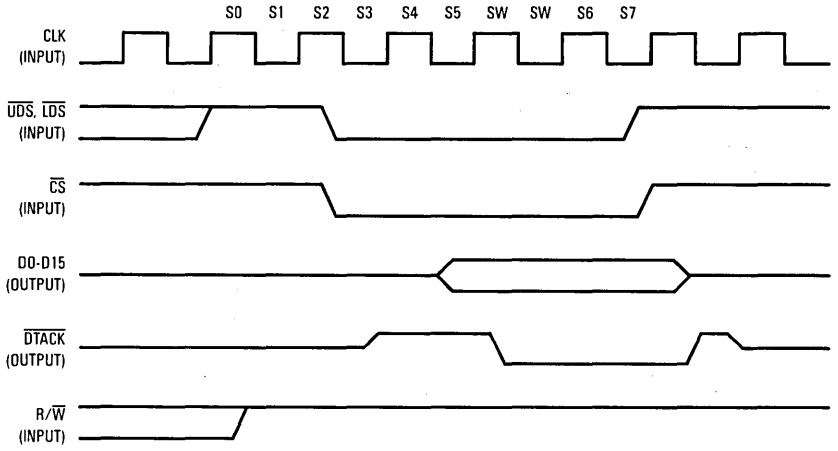


Figure 6-1. Host Processor Read Cycle with Two Wait States

or IV), is selected, the TBC accepts the data from the data bus and asserts DTACK. Next the TBC waits for either LDS/DS, UDS/A0, or CS to be negated, and then negates and three states DTACK and three states the data lines. The timing for this operation is shown in Figure 6-2.

6

6.1.3 Interrupt Acknowledge Cycles

During an interrupt acknowledge cycle, the host processor responds to an interrupt request from the TBC. The timing of an interrupt acknowledge cycle is identical to an odd byte read cycle, except that it is started by the assertion of the interrupt acknowledge (IACK) signal rather than chip select (CS). Note that chip select (CS) and interrupt acknowledge (IACK) are mutually exclusive signals and should not be asserted at the same time. If interrupt acknowledge (IACK) is asserted at the same time the TBC is the bus master, an address error is generated.

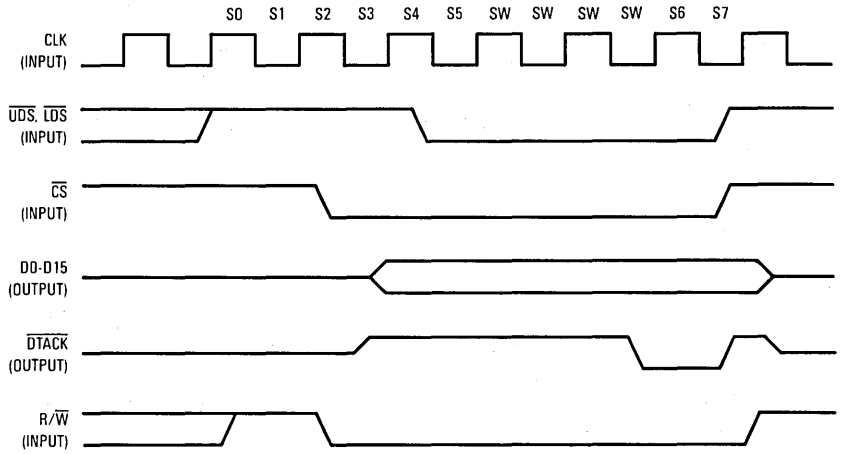


Figure 6-2. Host Processor Write Cycle

The interrupt acknowledge operation is started by the TBC when interrupt acknowledge ($\overline{IACK}$) and lower data strobe ($\overline{LDS}$) are asserted by the host processor. The TBC responds by placing its interrupt vector number on D0-D7 and asserting data transfer acknowledge ($\overline{DTACK}$). During this operation, A1-A31 and $\overline{UDS}$/A0 are ignored by the TBC. The vector number remains on the data bus until $\overline{IACK}$ or $\overline{LDS}$ is negated by the host processor. When this occurs, the TBC will three state the data lines and then negate and three state $\overline{DTACK}$. The timing for this operation is shown in Figure 6-3.

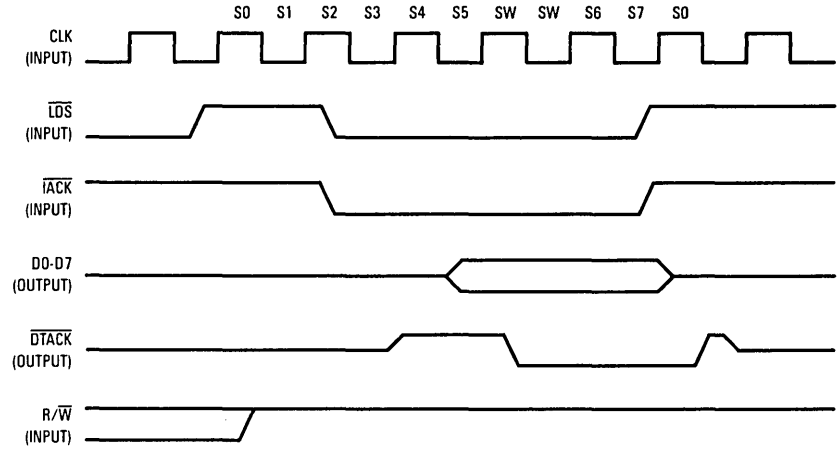


Figure 6-3. Interrupt Acknowledge Cycle

6.2 DMA OPERATION

In the DMA operation mode or master mode, the TBC is the bus master and performs memory read and write operations. The TBC can operate in either an 8-bit or a 16-bit bus configuration.

NOTE

The TBC does not check for an "odd pointer" and does not generate an address error for an odd word boundary. For an 8-bit data bus, an "odd pointer" is proper. For a 16-bit bus, the TBC zeros the least significant address bit and therefore always presents an even word address to the bus. The system designer must ensure that the TBC is not supplied with an "odd pointer" in a 16-bit configuration.

6.2.1 DMA Burst Control

The TBC has an operation mode that controls the maximum number of DMA transfers that the TBC can perform in one DMA burst. The operation mode bit is called halt generator enable (HLEN) and is controlled by the SET MODE 3 command. If the TBC is to be given the maximum memory bandwidth, HLEN should be set to zero. In this case the TBC will empty or fill the FIFO in one DMA burst. If the TBC is to have limited DMA burst, HLEN should be set to one. In this case, the TBC will release the bus after a maximum of eight memory cycles. If the FIFO is not empty or full, the TBC will request the bus again. This limited DMA burst mode provides an upper bound for latency and allows other masters to access the memory.

6.2.2 TBC Read Cycles

During a DMA read operation, the TBC controls the transfer of the data from memory into the TBC. The functional timing for this operation is shown in Figure 6-4. The TBC drives FC0-FC3 and A1-A31 pins with the address of the memory location that the TBC wants to read. $\overline{AS}$, and depending on the data size, $\overline{UDS}$/A0 and/or $\overline{LDS}$/$\overline{DS}$ are asserted and R/$\overline{W}$ is driven high. Data transfer acknowledge is asserted by the system when valid data from memory are on the appropriate data lines. If in the 8-bit bus mode, only the data on lines D0-D7 are assumed to be valid. When $\overline{DTACK}$ is asserted, data is latched by the TBC from the data lines, and the address, function code and control signals are negated.

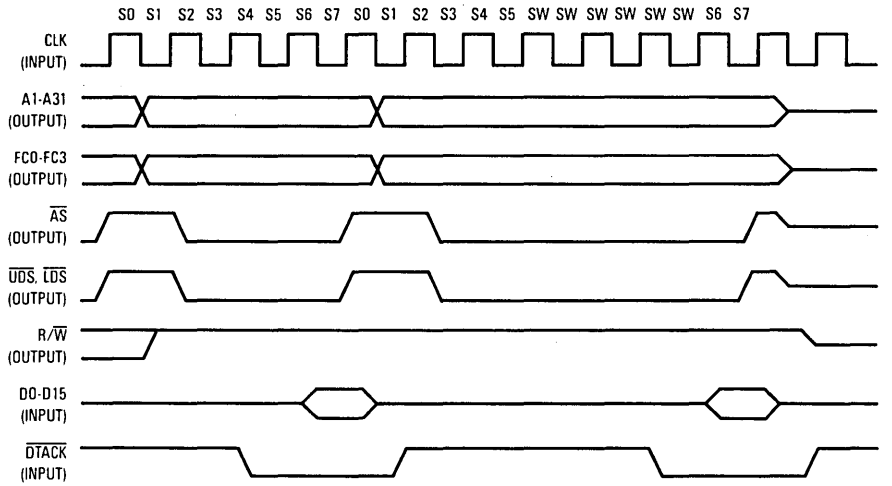


Figure 6-4. Read Cycle and Slow Read Cycle

6.2.3 TBC Write Cycles

During a DMA write operation, the TBC controls the transfer of data to memory from the TBC internal registers or FIFO. The functional timing for this operation is shown in Figure 6-5. The TBC drives FC0-FC3 and A1-A31 pins with the address of the memory location to be written. Then, address strobe ($\overline{AS}$) is asserted and, depending on the data size, upper data strobe ($\overline{UDS}$/A0) and lower data strobe ($\overline{LDS}$/$\overline{DS}$) are asserted and R/$\overline{W}$ is driven low. Data to be written to memory is placed on the bus and, when $\overline{DTACK}$ is asserted, the cycle is terminated. On the slow write cycle shown in Figure 6-5, the bus cycle is extended because $\overline{DTACK}$ does not appear as soon as expected by the TBC (by the end of S4).

6.3 BUS EXCEPTION CONTROL FUNCTIONS

To fully support the M68000 bus architecture, the TBC has three encoded inputs, $\overline{BEC0}$-$\overline{BEC2}$, that indicate abnormal bus cycle termination conditions. These three lines function in a similar manner to $\overline{RESET}$, $\overline{HALT}$, and $\overline{BERR}$ signals on a M68000 processor, but have different functionality than the processor counterparts. The 3-bit bus exception control code allows eight different bus termination conditions as shown in Figure 6-6. $\overline{BEC0}$ is the least significant bit and $\overline{BEC2}$ is the most significant bit. When an exception is encoded on the $\overline{BEC}$ lines, the TBC will terminate the current

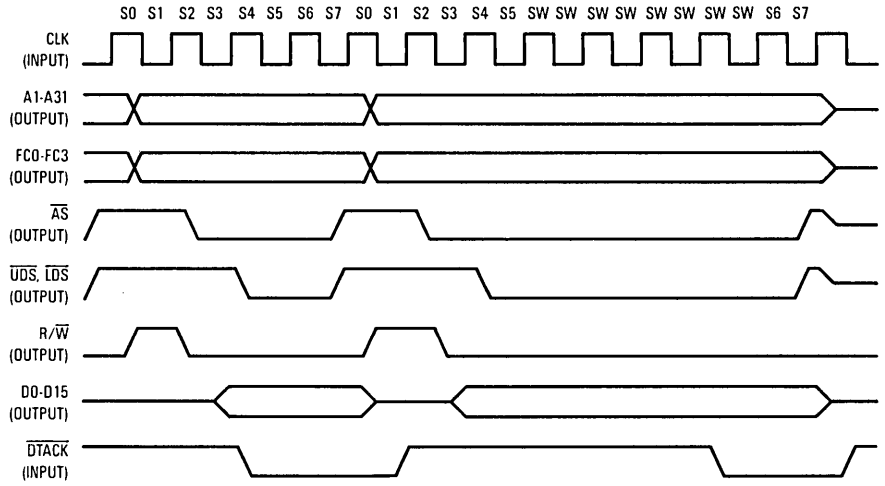


Figure 6-5. Write Cycle and Slow Write Cycle

Code	$\overline{BEC2}$	$\overline{BEC1}$	$\overline{BEC0}$	Definition	TBC Action
0	H	H	H	NO EXCEPTION	NO AFFECT
1	H	H	L	HALT (RELEASE BUS)	HALT AFTER DTACK AND RELEASE THE BUS
2	H	L	H	BUS ERROR	TERMINATE THE CURRENT CYCLE AND RELEASE THE BUS
3	H	L	L	RETRY	TERMINATE THE CURRENT CYCLE AND RERUN THE SAME CYCLE AGAIN AFTER THE EXCEPTION DISAPPEARS
4	L	H	H	RELINQUISH AND RETRY	TERMINATE THE CURRENT CYCLE, RELEASE THE BUS, RERUN LAST CYCLE AFTER REARBITRATION
5	L	H	L	UNDEFINED	ONE CLOCK DELAY BEFORE TERMINATION NO FURTHER CYCLES UNTIL NO EXCEPTION (RESERVED)
6	L	L	H	UNDEFINED	ONE CLOCK DELAY BEFORE TERMINATION NO FURTHER CYCLES UNTIL NO EXCEPTION (RESERVED)
7	L	L	L	RESET	RESET TBC REGISTERS AND LOGIC

Figure 6-6. $\overline{BEC}$ Encoding Definitions

bus cycle and wait for the $\overline{BEC}$ encoding to return to zero. Each of the possible conditions is discussed in detail in the following paragraphs.

There are three possible cases for bus exceptions. In the first case, a very early bus exception occurs when the bus exception is asserted more than one clock cycle before $\overline{DTACK}$ is asserted. The bus exception is acted on without any delay by the TBC and the bus cycle is terminated. In case two, which is the typical case, the bus exception occurs in the same clock cycle as $\overline{DTACK}$. One delay cycle is added to the bus cycle to be sure that a bus exception has occurred before the bus cycle is terminated. In the third case, the bus exception signal occurs on the clock after data transfer acknowledge ($\overline{DTACK}$) has been asserted. In this case, one clock delay is also added before the bus cycle is terminated.

6.3.1 Normal Termination

When the $\overline{BEC}$ pins are non-asserted, the TBC operates in a normal mode and $\overline{DTACK}$ terminates the bus cycle.

6.3.2 Halt

A logical one encoded on the $\overline{BEC}$ pins halts the TBC and the current bus cycle is terminated by the assertion of $\overline{DTACK}$. The TBC releases ownership of the bus and enters the idle state until the $\overline{BEC}$ pins return to normal mode (logical zero). The halt timing diagram is shown in Figure 6-7. The TBC rearbitrates for the bus and continues DMA operations if necessary.

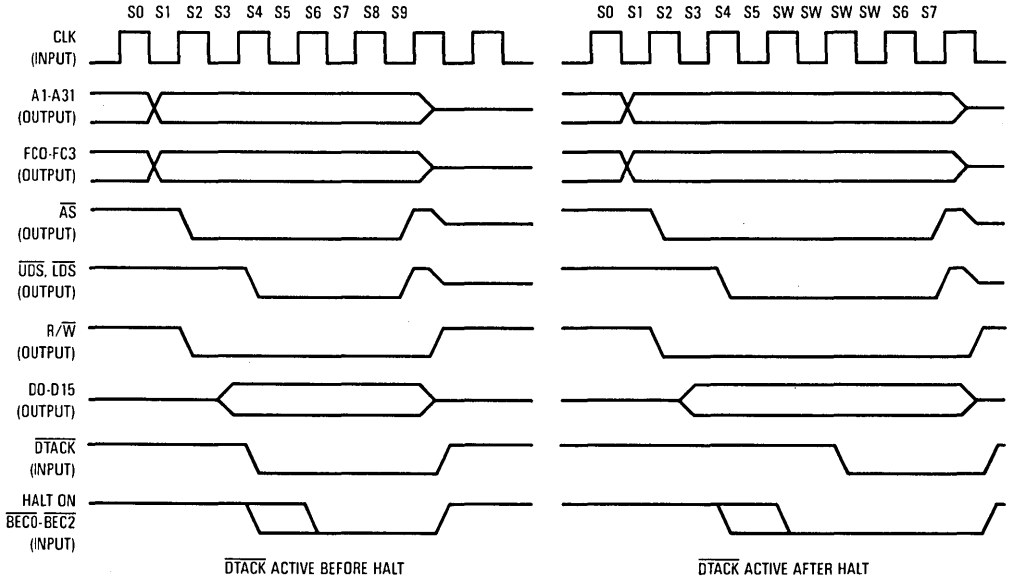


Figure 6-7. TBC Write Cycle with HALT

6.3.3 Bus Error

When a logical two is encoded on the $\overline{BEC}$ pins, the TBC aborts the current bus cycle and releases bus mastership. After the $\overline{BEC}$ lines return to normal, the TBC rearbitrates for the bus to report the bus error to the host processor by writing the address that caused the error into the DMA dump area in the initialization table. The TBC sets the bus/address error bit in the initialization table interrupt status word, and interrupts the host processor if enabled. A bus error condition is shown in the Figure 6-8 timing diagram.

6.3.4 Retry

A logical three encoded on the $\overline{BEC}$ pins of the TBC terminates the current bus cycle and places the TBC into a waiting mode. The TBC retains bus mastership by keeping $\overline{BGACK}$ asserted. When the $\overline{BEC}$ pins return to normal, the TBC reruns the same bus cycle using the same address and function code. Figure 6-9 shows the timing diagram for a retry operation.

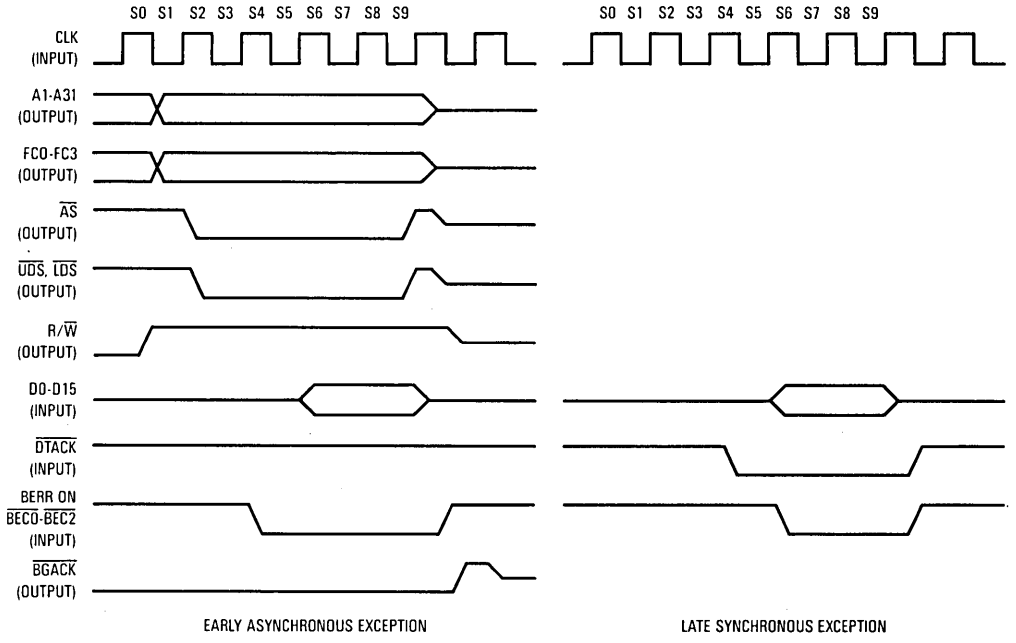


Figure 6-8. TBC Read Cycle with Bus Error

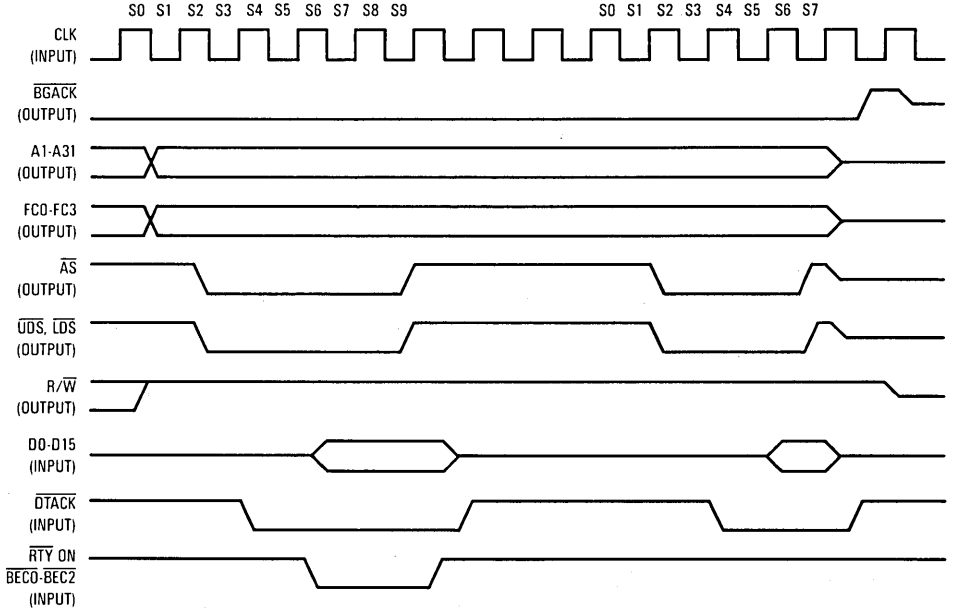


Figure 6-9. TBC Read Cycle with RETRY

6.3.5 Relinquish and Retry

If the $\overline{BEC}$ pins are encoded with a logical four, then the TBC terminates the current bus cycle and relinquishes bus mastership. One debounce delay after the $\overline{BEC}$ pins have returned to normal, the TBC rearbitrates for the bus and reruns the same bus cycle as shown in Figure 6-10.

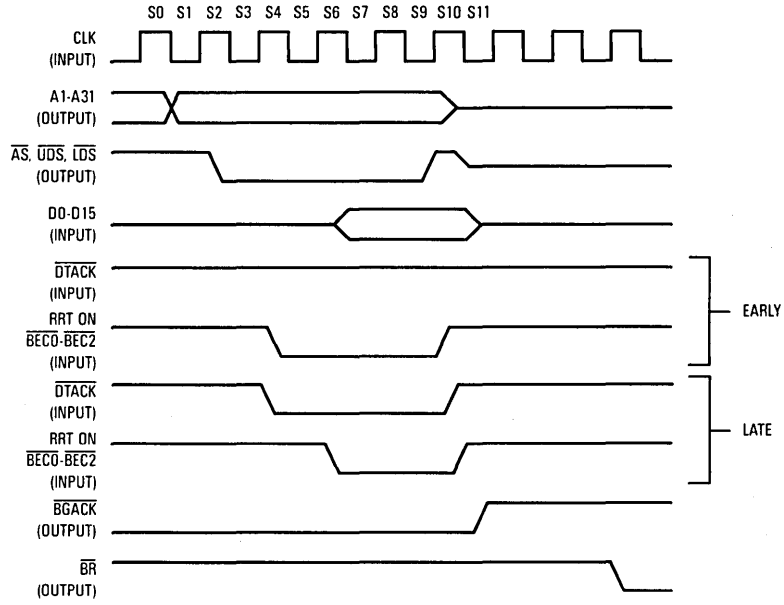


Figure 6-10. TBC Read Cycle with Relinquish and Retry, Early and Late

6.3.6 Reset

A logical seven encoded on the $\overline{BEC}$ pins causes the TBC to execute an internal reset sequence. Hardware and software reset will cause the TBC to execute the same steps, see **3.2.5 RESET Command** for more details.

6.3.7 Undefined $\overline{BEC}$ Codes

If a logical five or six is encoded on the $\overline{BEC}$ pins, the TBC takes no action as shown in Figure 6-11. However, the bus cycle may be extended one more clock cycle due to the debouncing of $\overline{BEC}$ lines. The TBC terminates the current cycle after $\overline{DTACK}$ is asserted.

6.4 BUS ARBITRATION

Bus arbitration is a technique used by bus master devices to request, be granted, and acknowledge bus mastership. In its simplest form, it consists of the following.

- Asserting a bus mastership request.
- Receiving a bus grant indicating that the bus is available at the end of the current cycle.

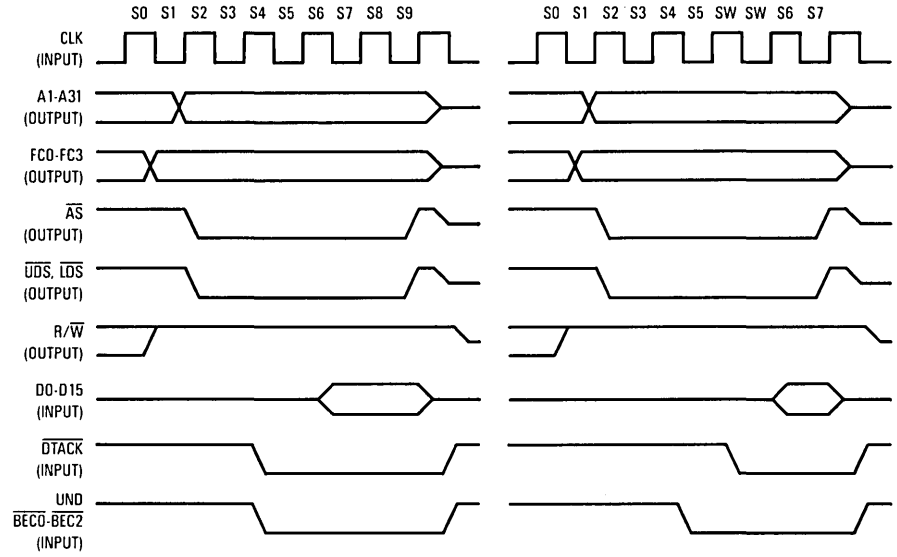


Figure 6-11. TBC Read Cycle with Unefined Bus Exception

- Monitoring the bus for the previous bus master to release the bus and acknowledging that bus mastership has been assumed.

The TBC uses this M68000 bus arbitration protocol to request bus mastership before entering the DMA mode of operation. The bus arbitration timing diagram is shown in Figure 6-12.

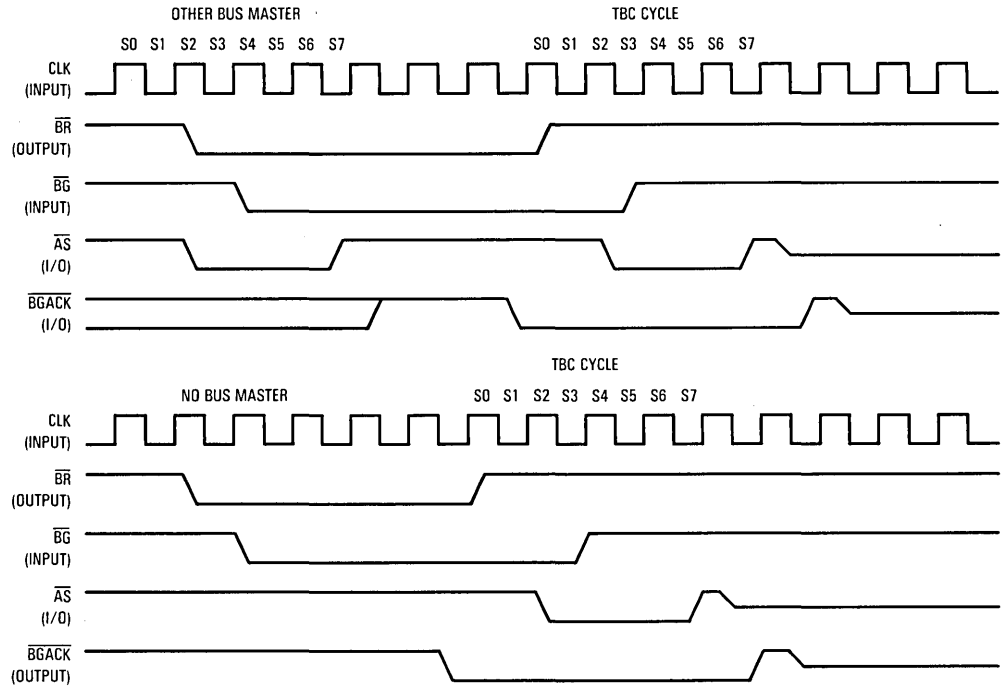


Figure 6-12. Bus Arbitration

6.4.1 Requesting the Bus

External devices capable of becoming bus masters request the bus by asserting the bus request ($\overline{BR}$) signal. This is a wire-ORed signal that indicates to the external bus arbiter that some external device requires control of the external bus. The bus is requested by the TBC when a command is issued that requires a DMA operation, or when data needs to be moved to or from the internal FIFO. The TBC will request the bus when there are four empty words in the FIFO in the transmit case, or when four words have been received in the FIFO in the receive case.

6.4.2 Receiving the Bus Grant

The host processor or a bus arbiter circuit asserts bus grant ($\overline{BG}$) as soon as possible to indicate that another bus master may assume control after the current bus master releases the bus. Normally this is immediately after internal synchronization of the host processor. The only exception to this occurs when the host processor has made an internal decision to execute the next bus cycle, but has not progressed far enough into the cycle to have asserted the address strobe ($\overline{AS}$) signal. In this case, bus grant will be delayed until $\overline{AS}$ is asserted to indicate that a bus cycle is being executed.

The bus grant ($\overline{BG}$) signal may be routed through a daisy-chained network or through a specific priority-encoded network. The host processor is not affected by the external method of arbitration as long as the protocol is obeyed.

6.4.3 Acknowledgement of Mastership

Upon receiving a bus grant, the TBC waits until $\overline{AS}$ and bus grant acknowledge ($\overline{BGACK}$) are negated before asserting $\overline{BGACK}$. The negation of $\overline{AS}$ indicates that the previous master completed its cycle; negation of $\overline{BGACK}$ indicates that the previous master has released control of the bus. When these conditions are met, the TBC asserts $\overline{BGACK}$. After $\overline{BGACK}$ is asserted, $\overline{BR}$ is negated to allow the external arbiter to begin arbitration of the next bus master. The current bus master only maintains control of the bus until it negates $\overline{BGACK}$.

6.4.4 Bus Arbitration Control

The bus arbitration control unit in the TBC is implemented with a finite state machine. The state diagram of this machine is shown in Figure 6-13. All asynchronous signals to the TBC are synchronized before they are used internally. This synchronization is accomplished in a maximum of one cycle of the system clock. The input signal is sampled on the falling edge of the clock and is valid internally after the next rising edge.

6.4.5 Reset Operation

The TBC is reset either by a host processor command passed to the TBC or by a hardware reset from an external device. The host processor can issue a reset by passing the TBC a RESET command (hex "FF"). The hardware reset is accomplished by encoding reset on $\overline{BEC0}$-$\overline{BEC2}$. A reset encoding on the $\overline{BEC}$ lines should be asserted for at least ten clock cycles. Normally, the TBC requires four clock cycles before it is ready to accept a subsequent command after either a hardware or software reset.

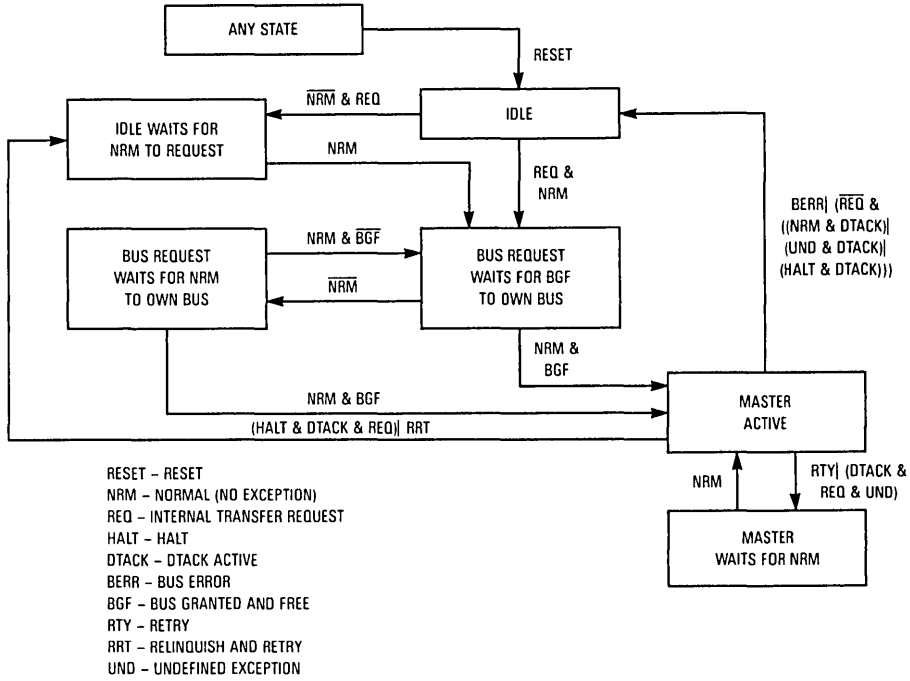


Figure 6-13. Bus Arbitration Unit State Diagram

6.5 BUS OVERHEAD TIME

In asynchronous bus systems, such as those defined for the M68000 Family, a certain amount of time is used to synchronize incoming signals. This is "wasted time" since no data transfer activity can take place during those periods. In many applications, the synchronization overhead time required to switch bus masters must be known to predict system behavior.

For the TBC, there are two types of overhead:

Time for the TBC to take control of the bus (front-end overhead), and

Time that occurs when the TBC releases control of the bus to another bus master (back-end overhead).

The timing diagram for the front-end and back-end overhead for the TBC is shown in Figure 6-14.

6.5.1 Front-End Overhead

This overhead is the delay that occurs from the time that the host processor terminates a bus cycle by negating $\overline{AS}$ to when the TBC starts the bus cycle by placing the function codes and address information on the bus. It is assumed that $\overline{BG}$ is asserted and $\overline{BGACK}$ is negated prior to negation of $\overline{AS}$ by the host processor so that no additional synchronization delays are introduced by those signals. After one synchronization delay plus one and a half clock cycles, the TBC asserts $\overline{BGACK}$ to assume control of the bus and begin the DMA cycle. The front-end overhead is between two-and-a-half and three-and-a-half clock cycles.

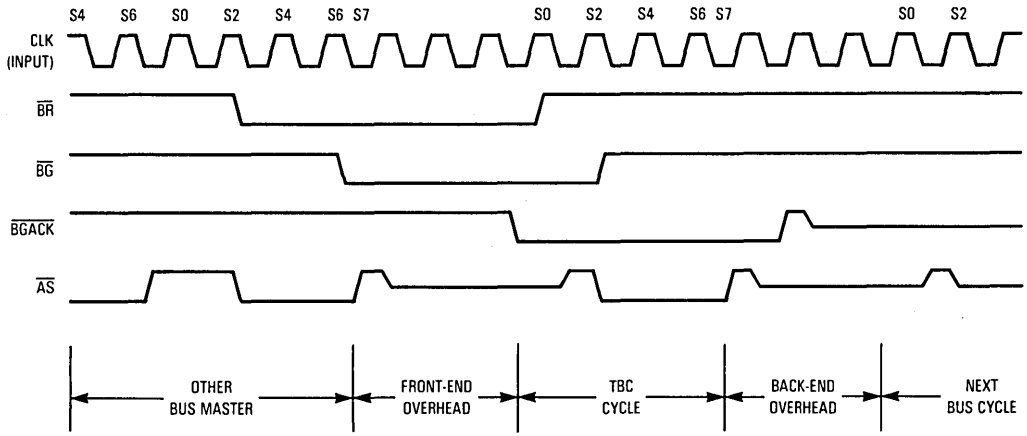


Figure 6-14. Bus Timing Diagram

6.5.2 Back-End Overhead

6

This overhead time is the delay between when the TBC has completed all operations and negated $\overline{AS}$, and the start of the next bus cycle which is controlled by another bus master. The TBC negates the $\overline{BGACK}$ signal one cycle after the last bus cycle. One synchronization delay plus one-half clock cycle later, the new bus master begins the next bus cycle.

6.6 REGISTERS

The following registers are user programmable: the command register (CR), the data register (DR), and the interrupt vector register (IV). Table 6-1 shows how the TBC registers are accessed for 8-bit bus and Table 6-2 shows how the TBC registers are accessed using a 16-bit data bus.

Table 6-1. 8-Bit Bus Access ($\overline{CS}$=0 and R/$\overline{W}$=0)

A2	A1	A0	$\overline{DS}$	TBC Register
0	0	X	0	CR
0	1	X	0	IV
1	0	0	0	DR — Byte 3
1	0	1	0	DR — Byte 2
1	1	0	0	DR — Byte 1
1	1	1	0	DR — Byte 0

Table 6-2. 16-Bit Bus Access ($\overline{CS}$=0 and R/$\overline{W}$=0)

A2	A1	$\overline{UDS}$	$\overline{LDS}$	TBC Register
0	0	X	0	CR
0	1	X	0	IV
1	0	0	1	DR — Byte 3
1	0	1	0	DR — Byte 2
1	0	0	0	DR — High Word
1	1	0	1	DR — Byte 1
1	1	1	0	DR — Byte 0
1	1	0	0	DR — Low Word

SECTION 7
TBC INTERFACES

The TBC must be interfaced to a host processor, buffer memory, and the physical layer of the communication medium.

7.1 TBC-TO-HOST PROCESSOR INTERFACE

Figure 7-1 illustrates the connection of the TBC to a MC68020 host processor, while Figure 7-2 illustratres the interface of the TBC to iAPX80186. For detailed descriptions of the bus signals see **SECTION 5 SIGNALS.**

7.2 NON-M68000 BUS INTERFACE

The TBC has a programmable mode which allows use with non-Motorola byte ordered memory structures. Figure 7-3 illustrates, for example, the comparison between the Intel 8086 and the Motorola M68000 memory configurations.

Information stored in a data buffer is assumed to be organized in bytes. The first byte is byte 0 and the last byte is byte 4. The TBC can internally swap the data on the bus according to the swap bit in the SET MODE 3 command during TBC initialization. The byte swapping option is limited to data that is transmitted or received in the data buffers. Initialization table entries and pointers and control data in frame descriptors *MUST* be organized to the Motorola standard.

Another consideration besides byte order is control line usage. In case of the Intel 8086 Family, the address lines always provide a full byte address using A0. A signal called $\overline{\text{BHEN}}$ is used to indicate a data transfer on D8-D15. As a simple interface to Intel type devices, the TBC signals can be utilized as follows: $\overline{\text{UDS}}$ is used as $\overline{\text{BHEN}}$, $\overline{\text{LDS}}$ is used as A0. In a similiar manner, it is possible to map bus arbitration schemes. In this case, the Intel $\overline{\text{HOLD}}$ function is similiar to Motorola bus request, and HOLDA can be treated as bus grant. $\overline{\text{BGACK}}$ also provides extra information to the system. In cases where the TBC may be enabled for unlimited DMA burst cycles, the user may want to create a capability for the processor to request the bus immediately. This can be accomplished in a potentially disruptive manner through encoding BERR on the $\overline{\text{BEC}}$ lines, or in a more predictable manner through HALT or RETRY encodings.

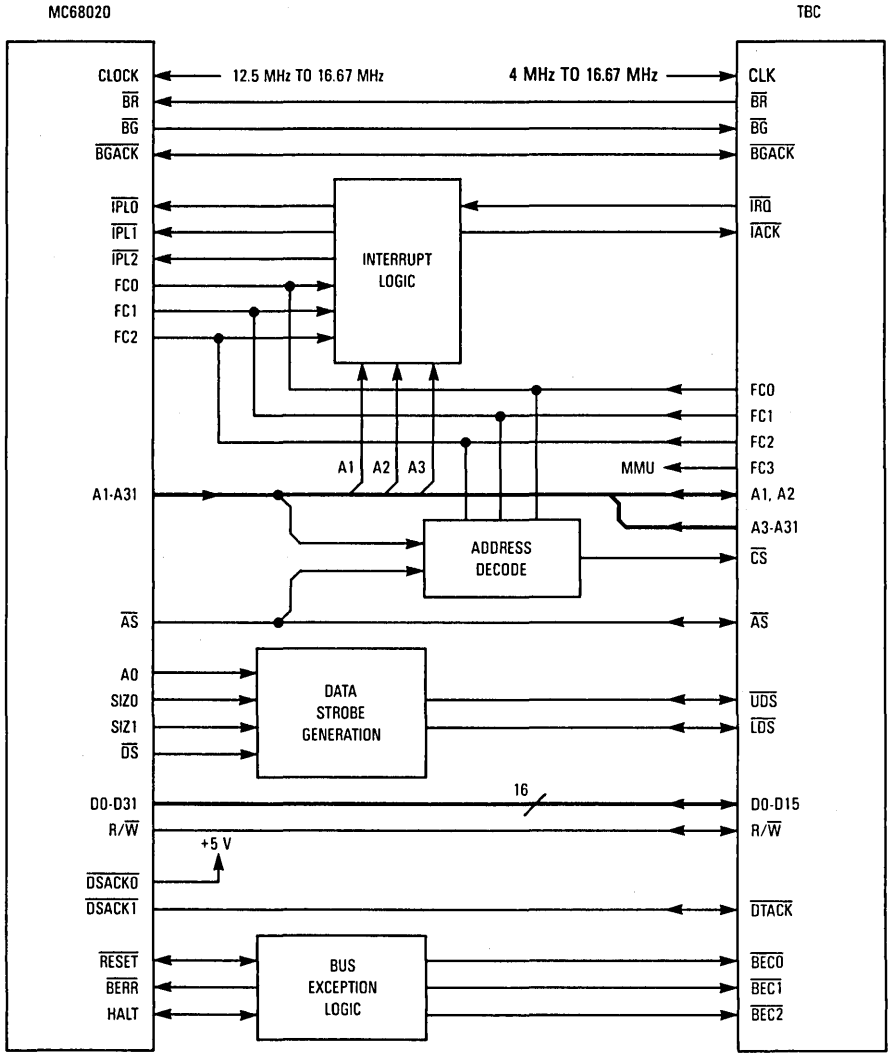


Figure 7-1. TBC to MC68020 Interface

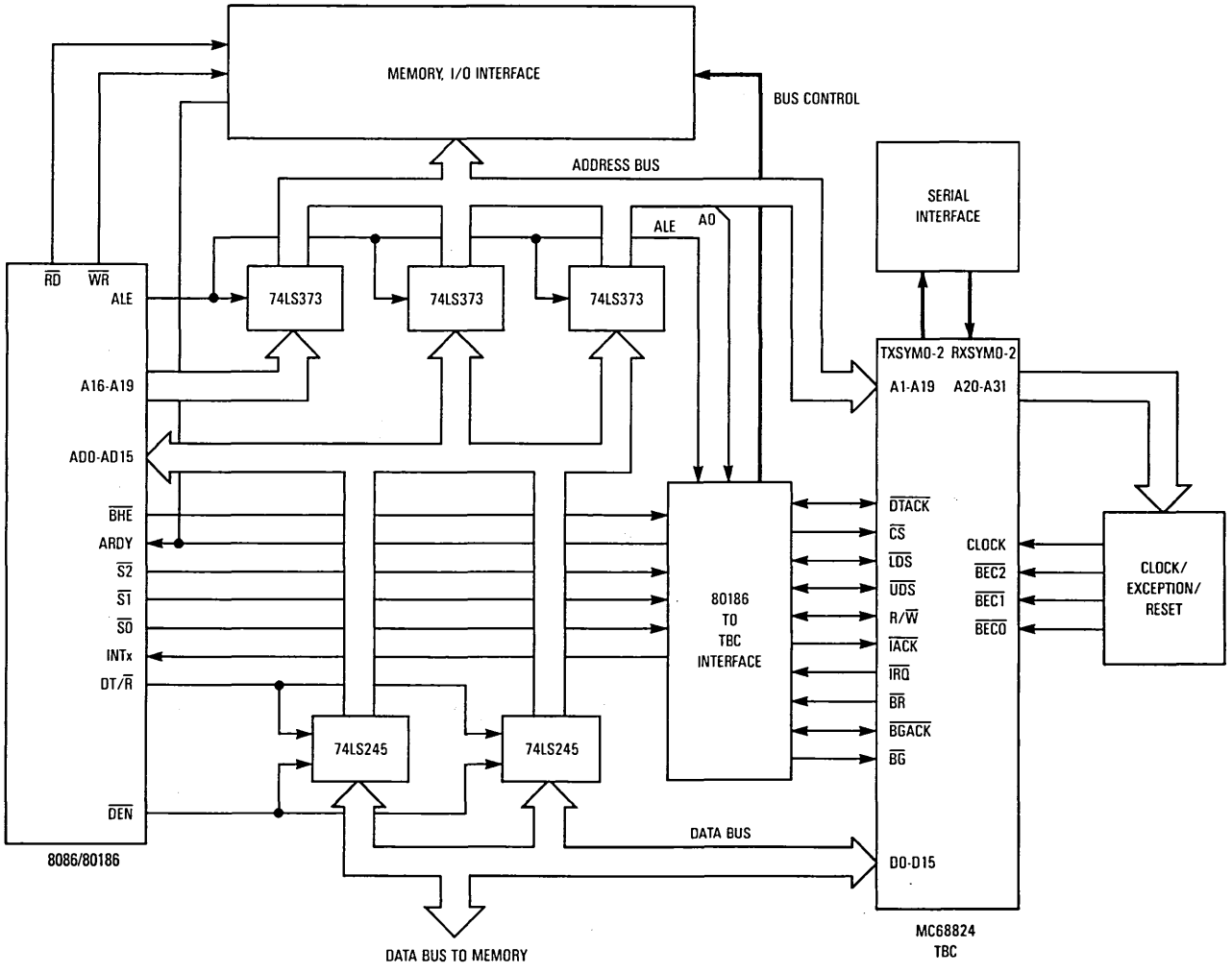


Figure 7-2. TBC to iAPX 80186 Interface

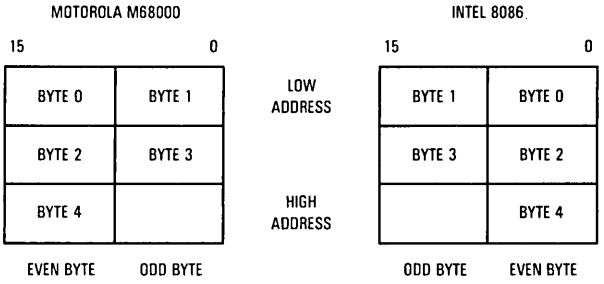


Figure 7-3. Memory Organization

7.3 MAC SUBLAYER-TO-PHYSICAL-LAYER INTERFACE

This physical layer interface complies with the IEEE Draft Standard 802.4G. Physical data units are sent and received as atomic symbols. To minimize the number of signal lines, the station management functions required by the physical layer make use of the data lines. The three primary functions of the interface are:

Physical Data Request Channel
In the MAC mode, this channel is used by the TBC to provide encoded requests for data transmission.

Physical Data Indication Channel
In the MAC mode, this channel is used by the physical layer to provide encoded indications of data unit reception to the TBC.

Physical Layer Management
In the station management mode, this interface also provides the ability to pass station management requests, indications, and confirmations between the MAC device and the physical layer. Although management is not directly a MAC function, the MAC interface is used as a channel to provide this service. The request channel consists of four basic signal lines. These signal lines are synchronized to the transmit clock (TXCLK). The indication channel also consists of four signals which are synchronized to the receive clock (RXCLK).

See **5.6 SERIAL INTERFACE** for a detailed description of the serial interface.

SECTION 8
ELECTRICAL SPECIFICATIONS

This section contains the electrical specifications and associated timing information for the MC68824.

8.1 MAXIMUM RATINGS

Rating	Symbol	Value	Unit
Supply Voltage	V_{DD}	−0.3 to +7.0	V
Input Voltage	V_{in}	−0.3 to +7.0	V
Operating Temperature MC68824 MC68824I	T_A	 0 to 70 0 to 85	°C
Storage Temperature	T_{stg}	−55 to 150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum-rated voltages to this high-impedance circuit. Reliability of operation is enhanced if unused inputs are tied to an appropriate logic voltage level (e.g., either GND or V_{DD}).

8.2 THERMAL CHARACTERISTICS

Characteristic	Symbol	Value	Unit
Thermal Resistance for PGA	θ_{JA}	33	°C/W

$T_J = T_A + (P_D \cdot \theta_{JA})$
$P_D = (V_{DD} \cdot I_{DD}) + P_{I/O}$
where:
$P_{I/O}$ is the lower dissipation on pins (user determined) which can be neglected in most cases.

For $T_A = 70°C$ and $P_D = 0.55$ W @ 12.5 MHz
$T_J = 88°C$

8

8.3 POWER CONSIDERATIONS

The average chip-junction temperature, T_J, in °C can be obtained from:

$$T_J = T_A + (P_D \cdot \theta_{JA}) \quad (1)$$

where:
T_A = Ambient Temperature, °C
θ_{JA} = Package Thermal Resistance, Junction-to-Ambient, °C/W
P_D = $P_{INT} + P_{I/O}$
P_{INT} = $I_{DD} \times V_{DD}$, Watts — Chip Internal Power
$P_{I/O}$ = Power Dissipation on Input and Output Pins, Watts — User Determined

For most applications $P_{I/O} < P_{INT}$ and can be neglected.

The following is an approximate relationship between P_D and T_J (if $P_{I/O}$ is neglected):

$$P_D = K \div (T_J + 273°C) \quad (2)$$

Solving equations (1) and (2) for K gives:

$$K = P_D \cdot (T_A + 273°C) + \theta_{JA} \cdot P_D^2 \quad (3)$$

where K is a constant pertaining to the particular part. K can be determined from equation (3) by measuring P_D (at equilibrium) for a known T_A. Using this value of K, the values of P_D and T_J can be obtained by solving equations (1) and (2) iteratively for any value of T_A.

8.4 DC ELECTRICAL CHARACTERISTICS

All specifications are valid under the following conditions: V_{DD} = 4.75 V to 5.25 V, V_{SS} = 0 V, $T_A = T_L$ to T_H and 130 pF total capacitance on output pins.

Characteristic		Symbol	Min	Max	Unit
Input High Voltage (Except System Clock)		V_{IH}	2.0	V_{DD}	V
Input Low Voltge (Except System Clock)		V_{IL}	$V_{SS}-0.3$	0.8	V
Input High Voltage (System Clock)		V_{CIH}	2.4	V_{DD}	V
Input Low Voltage (System Clock)		V_{CIL}	$V_{SS}-0.3$	0.5	V
Input Leakage Current @5.25 V		I_{in}	—	20	μA
Input Capacitance (V_{in} = 0 V, T_A = 25°C, F = 1 MHz)		C_{in}	—	13	pF
Three-State Leakage Current @2.4/0.5 V		I_{TSI}	—	20	μA
Open-Drain Leakage Current @2.4 V		I_{OD}	—	20	μA
Output High Voltage		V_{OH}			V
(I_{OH} = 1 mA)	$\overline{SMREQ}$, TXSYM2, TXSYM1, TXSYM0		2.5	—	
(I_{OH} = 400 μA)	All Others		2.4	—	
Output Low Voltage		V_{OL}			V
(I_{OL} = 8.0 mA)	$\overline{SMREQ}$, TXSYM2, TXSYM1, TXSYM0		—	0.5	
(I_{OL} = 3.2 mA)	A1-A31, FC0-FC3, and $\overline{UDS}$/A0 as A0		—	0.5	
(I_{OL} = 5.3 mA)	D0-D15, $\overline{AS}$, $\overline{LDS}$/DS, $\overline{UDS}$/A0 as $\overline{UDS}$, $\overline{DTACK}$, $\overline{BGACK}$, R/$\overline{W}$		—	0.5	
(I_{OL} = 8.9 mA)	$\overline{IRQ}$, $\overline{BR}$		—	0.5	
Power Dissipation	@ 10 MHz, 0°C	P_D	—	0.50	W
	@ 12.5 MHz, 0°C			0.55	
	@ 16.67 MHz, 0°C			0.70	

8.5 AC ELECTRICAL CHARACTERISTICS

High and low outputs are measured at 2.0 V minimum and 0.8 V maximum respectively, except $\overline{SMREQ}$ and TXSYM0-2 which are measured from 2.5 V and 0.5 V. High and low inputs are driven to 2.4 V and 0.5 V respectively for AC test purposes. However, input specifications are still measured from 2.0 V to 0.8 V. All specifications are valid under the following conditions: (V_{DD} = 4.75 V to 5.25 V, V_{SS} = 0 V, $T_A = T_L$ to T_H, output load = 130 pF, and output current as specified in **8.4 DC ELECTRICAL CHARACTERISTICS**)

Num.	Characteristic	10 MHz		12.5 MHz		16.67 MHz		Unit
		Min	Max	Min	Max	Min	Max	
1	Asynchronous Input Setup Time	20	—	20	—	10	—	ns
2	$\overline{UDS}$, $\overline{LDS}$ Inactive to $\overline{CS}$, $\overline{IACK}$ Inactive	—	100	—	80	—	60	ns
3	CLK Low (On Which $\overline{UDS}$ or $\overline{LDS}$ and $\overline{CS}$ or $\overline{IACK}$ are Recognized) to Data-Out Valid (see Note 5)	— —	1/2 +150	— —	1/2 +120	— —	1/2 +90	Clk. Per. ns
4	$\overline{CS}$ or $\overline{IACK}$ High to Data-Out High-Impedance	—	60	—	50	—	35	ns
5	$\overline{LDS}$/DS High to Data-Out Hold Time (see Note 6)	0	—	0	—	0	—	ns
6	$\overline{IACK}$ or $\overline{CS}$ Low to $\overline{DTACK}$ High (Driving Three-State $\overline{DTACK}$ High)	—	80	—	70	—	60	ns
7	CLK Low (On Which $\overline{UDS}$ or $\overline{LDS}$ and $\overline{CS}$ or $\overline{IACK}$ are Recognized) to $\overline{DTACK}$ Low (see Note 5)	— —	2 +90	— —	2 +80	— —	2 +60	Clk. Per. ns
8	CLK Low to $\overline{DTACK}$ Low	—	90	—	80	—	50	ns
9	Data-Out Valid to $\overline{DTACK}$ Low	20	—	20	—	20	—	ns
10	$\overline{DTACK}$ Low to $\overline{UDS}$, $\overline{LDS}$, $\overline{CS}$, $\overline{IACK}$ High (Earliest)	100	—	80	—	60	—	ns
11	$\overline{CS}$ or $\overline{IACK}$ or Data Strobes (The Earliest) High to $\overline{DTACK}$ High (see Note 7)	—	60	—	50	—	40	ns
12	$\overline{DTACK}$ High to $\overline{DTACK}$ High Impedance (At End of Bus Cycle)	—	50	—	50	—	40	ns

— CONTINUED —

8.5 AC ELECTRICAL CHARACTERISTICS (Continued)

Num.	Characteristic	10 MHz		12.5 MHz		16.67 MHz		Unit
		Min	Max	Min	Max	Min	Max	
13	$\overline{UDS}$, $\overline{LDS}$ Inactive Time	1	—	1	—	1	—	Clk. Per.
14	$\overline{CS}$, $\overline{IACK}$ Inactive Time	0	—	0	—	0	—	ns
15	A1-A2 Valid to $\overline{UDS}$, $\overline{LDS}$, $\overline{CS}$ (The Latest One) Low (Write)	30	—	20	—	20	—	ns
16	$\overline{DTACK}$ Low to Data and A1-A2 Hold Time	100	—	80	—	60	—	ns
17	$\overline{UDS}$ or $\overline{LDS}$, $\overline{CS}$ or $\overline{IACK}$ (The Latest One) Low to Data-In Valid	—	80	—	70	—	60	ns
18	R/$\overline{W}$ Valid to $\overline{UDS}$, or $\overline{LDS}$, $\overline{CS}$ or $\overline{IACK}$ (The Latest One) Low	20	—	20	—	10	—	ns
19	$\overline{UDS}$, $\overline{LDS}$ High to R/$\overline{W}$ High	0	—	0	—	0	—	ns
20	CLK High to $\overline{IRQ}$ Low	—	100	—	80	—	60	ns
21	Reserved							
22	Reserved							
23	CLK High to $\overline{BR}$ Low	—	60	—	55	—	40	ns
24	CLK High to $\overline{BR}$ High Impedance	—	55	—	50	—	40	ns
25	$\overline{BGACK}$ Low to $\overline{BR}$ High Impedance	20	—	20	—	10	—	ns
26	$\overline{BG}$ Active/Inactive to CLK Low Setup Time	20	—	20	—	10	—	ns
27	CLK Low to $\overline{BGACK}$ Low	—	60	—	55	—	40	ns
28	CLK High to $\overline{BGACK}$ High Impedance	—	45	—	40	—	30	ns
29	$\overline{AS}$ and $\overline{BGACK}$ High, the Latest One, to $\overline{BGACK}$ Low (when $\overline{BG}$ is Previously Asserted)	2 +20	3 +80	2 +20	3 +70	2 +10	3 +50	Clk. Per. ns
30	$\overline{BG}$ Low to $\overline{BGACK}$ Low (No Other Bus Master)	2 +20	3 +80	2 +20	3 +70	2 +10	3 +50	Clk. Per. ns
31	$\overline{BR}$ High Impedance to $\overline{BG}$ High	0	—	0	—	0	—	ns
32	Clock on which $\overline{BGACK}$ Low to Clock on which $\overline{AS}$ Low	1.5	1.5	1.5	1.5	1.5	1.5	Clk. Per.
33	Clock Low to $\overline{BGACK}$ High	—	55	—	50	—	40	ns
34	CLK on which $\overline{BR}$ Low to CLK on which $\overline{BGACK}$ Low (Assuming that $\overline{BG}$ is Active and $\overline{BGACK}$ and $\overline{AS}$ are Inactive for at Least 2 CLK Periods)	1.5	1.5	1.5	1.5	1.5	1.5	Clk. Per.
35	CLK on which $\overline{AS}$ is High to CLK on which $\overline{BGACK}$ is High	—	1	—	1	—	1	Clk. Per.
36	CLK High to Address Valid	—	100	—	80	—	60	ns
37	CLK High to Address/FC High Impedance	—	70	—	60	—	50	ns
38	CLK High to FC Valid	—	60	—	55	—	50	ns
39	Address Valid to $\overline{AS}$ Valid	20	—	15	—	10	—	ns
40	CLK High to $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ Low	—	50	—	40	—	30	ns
41	CLK to $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High	—	55	—	50	—	45	ns
42	$\overline{AS}$ High to Address/FC Invalid	20	—	10	—	10	—	ns
43	CLK High to $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High Impedance	—	70	—	60	—	45	ns
44	CLK to R/$\overline{W}$ High (see Note 4)	—	55	—	50	—	45	ns
45	CLK Low to R/$\overline{W}$ High Impedance	—	70	—	60	—	45	ns
46	$\overline{UDS}$, $\overline{LDS}$ High to Data-In Invalid	0	—	0	—	0	—	ns
47	$\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High to $\overline{DTACK}$ High (Earliest of $\overline{AS}$, $\overline{UDS}$, or $\overline{LDS}$)	0	100	0	90	0	60	ns
48	Data-In to CLK Low Setup Time Required when $\overline{DTACK}$ Satisfies (1) (see Note 1)	10	—	10	—	5	—	ns

— CONTINUED —

8.5 AC ELECTRICAL CHARACTERISTICS (Continued)

Num.	Characteristic	10 MHz		12.5 MHz		16.67 MHz		Unit
		Min	Max	Min	Max	Min	Max	
49	$\overline{DTACK}$ Low to Data-In Valid Required when $\overline{DTACK}$ does not Satisfy (1) (see Note 2)	—	65	—	50	—	40	ns
50	CLK High to R/$\overline{W}$ Low	—	60	—	55	—	40	ns
51	$\overline{AS}$ Low to Data-Out Valid (Write)	—	90	—	80	—	60	ns
52	CLK Low to Data-Out Valid	—	55	—	55	—	40	ns
53	Data-Out Valid to $\overline{UDS}$, $\overline{LDS}$ Low	20	—	15	—	10	—	ns
54	$\overline{UDS}$, $\overline{LDS}$ High to Data-Out Invalid	20	—	15	—	10	—	ns
55	CLK High to Data-Out Hold Time	0	100	0	80	0	60	ns
56	No Exception to $\overline{BR}$ ($\overline{DTACK}$ Active)	1.5 +20	2.5 +80	1.5 +20	2.5 +70	1.5 +10	2.5 +50	Clk. Per. ns
57	$\overline{DTACK}$ Low to Asynchronous Exception Active Required when $\overline{DTACK}$ Does Not Satisfy (1) (see Note 2)	—	55	—	35	—	30	ns
58	Exception Active to CLK Low Setup Time Synchronous Input ("Late Exception") Required when $\overline{DTACK}$ Satisfies (1) (see Note 1)	45	—	45	—	20	—	ns
59	Exception Active to CLK Low Setup Time Asynchronous Input (Required when $\overline{DTACK}$ is Absent) (see Note 3)	20	—	20	—	10	—	ns
60	$\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ High to Exception Inactive	0	—	0	—	0	—	ns
61	Exception Inactive to CLK Low Setup Time (for Identification of No Exception)	20	—	20	—	10	—	ns
62	No Exception to $\overline{BR}$ ($\overline{DTACK}$ Inactive)	2.5 +20	3.5 +80	2.5 +20	3.5 +70	2.5 +10	3.5 +50	Clk. Per. ns
63	$\overline{RESET}$ (on $\overline{BEC0}$-$\overline{BEC2}$) Width	10	—	10	—	10	—	Clk. Per.
64	CLK Frequency	4	10	4	12.5	8	16.67	MHz
65	CLK Period	100	250	80	250	60	125	ns
66	CLK Width High (see Note 8)	45	125	35	125	25	62.5	ns
67	CLK Rise/Fall Time (see Note 8)	—	10	—	5	—	5	ns
68	CLK Width Low (see Note 8)	45	125	35	125	25	62.5	ns
69*	RXCLK, TXCLK Frequency	1	10	1	12.5	5	16.67	MHz
70	RXD Signals Setup Time	35	—	35	—	25	—	ns
71	RXD Signals Hold Time	5	—	5	—	5	—	ns
72	RXCLK, TXCLK Rise/Fall Time	—	10	—	10	—	5	ns
73*	RXCLK, TXCLK Width Low	40	525	40	525	25	100	ns
74*	RXCLK, TXCLK Width High	40	525	40	525	25	100	ns
75*	RXCLK, TXCLK Period	95	1050	80	1050	60	200	ns
76	TXCLK High to TXD Signals Output Delay	5	55	5	55	5	45	ns

*Parts with an S suffix have the following characteristics:

69	RXCLK, TXCLK Frequency	.01	10	.01	12.5	—	—	MHz
73	RXCLK, TXCLK Width Low	40	50,000	40	50,000	—	—	ns
74	RXCLK, TXCLK Width High	40	50,000	40	50,000	—	—	ns
75	RXCLK, TXCLK Period	95	100,000	80	100,000	—	—	ns

— CONTINUED —

8.5 AC ELECTRICAL CHARACTERISTICS (Concluded)

NOTES:

1. If $\overline{\text{DTACK}}$ satisfies the asynchronous setup time (1), then (48) is required for the data-in setup time and (58) for the synchronous exception setup time. Erroneous behavior may occur if (58) is not satisfied.
2. If $\overline{\text{DTACK}}$ does not satisfy (1), then (49) is required for data-in and (57) for the exception. Erroneous behaviour may occur if (57) is not satisfied.
3. Active exception when $\overline{\text{DTACK}}$ is absent must satisfy the asynchronous setup time (59).
4. R/$\overline{\text{W}}$ rises on the end of a write cycle (i.e., on the phase following S7). If the TBC relinquishes the bus, then R/$\overline{\text{W}}$ is three-stated one phase later. When the TBC takes the bus, R/$\overline{\text{W}}$ is three-stated until S1 and rises on that phase.
5. Data (3) and $\overline{\text{DTACK}}$ (7) will be timed from the earliest clock on which $\overline{\text{CS}}$ and either data strobe are recognized during an MPU cycle. Data (3) and $\overline{\text{DTACK}}$ will be timed from the earliest clock on which $\overline{\text{IACK}}$ and either data strobe during an $\overline{\text{IACK}}$ cycle.
6. If $\overline{\text{CS}}$ or $\overline{\text{IACK}}$ is negated before $\overline{\text{UDS}}$/$\overline{\text{LDS}}$, the data bus will be three-stated (4), possibly before $\overline{\text{UDS}}$/$\overline{\text{LDS}}$ negation.
7. If an 8-bit bus is used only $\overline{\text{LDS}}$ need be considered. If a 16-bit bus is used, both $\overline{\text{UDS}}$ and $\overline{\text{LDS}}$ must negate to apply to this specification.
8. The clock signal used during test has 5 ns of rise time and 5 ns of fall time. For system implementations that have less clock rise and fall time, the clock pulse minimum should be commensurately wider such that:
   1. System (TCL + (TCR + TCF) ÷ 2) ≥ (minimum TCYC) ÷ 2
   2. System (TCH + (TCR + TCF) ÷ 2) ≥ (minimum TCYC) ÷ 2

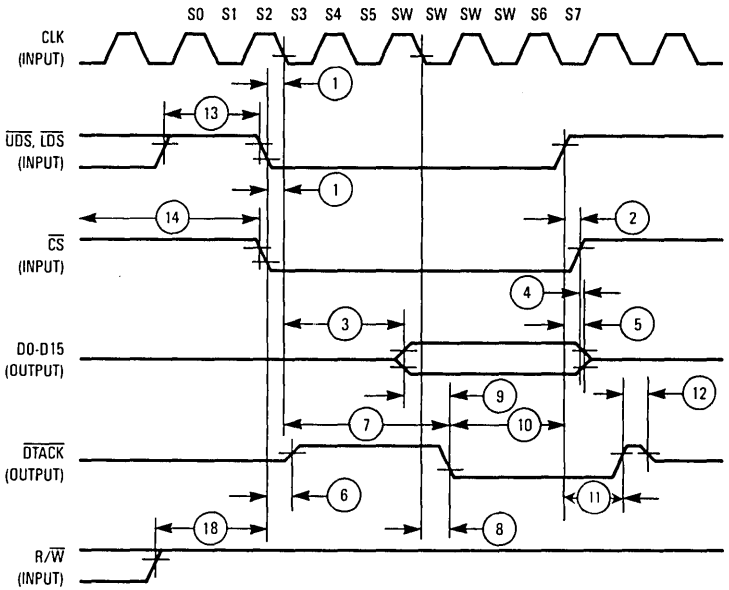


Figure 8-1. Host Processor Read Cycle

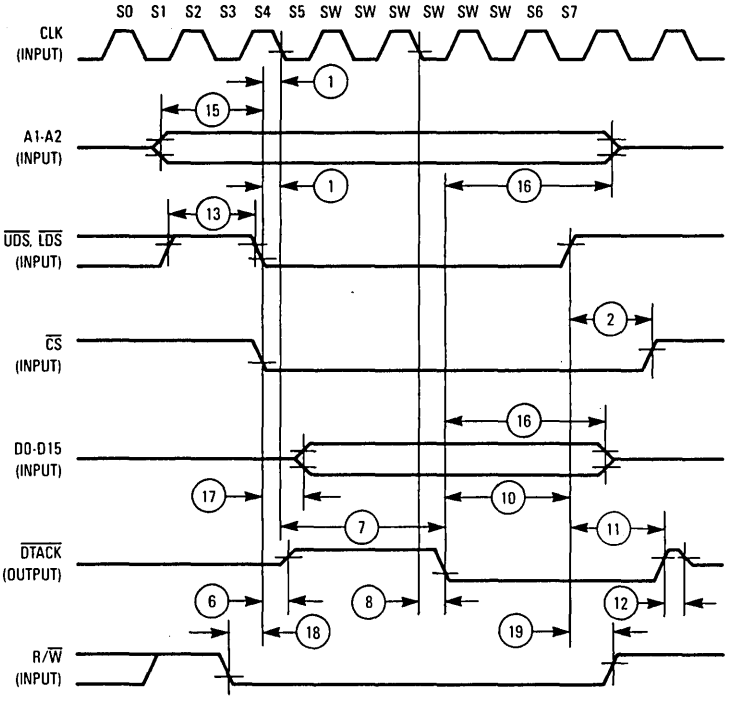


Figure 8-2. Host Processor Write Cycle

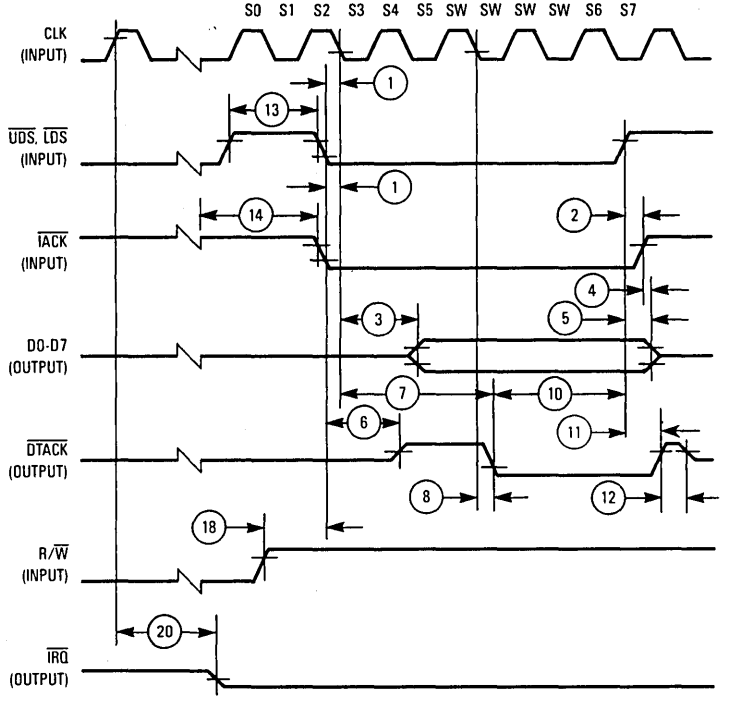
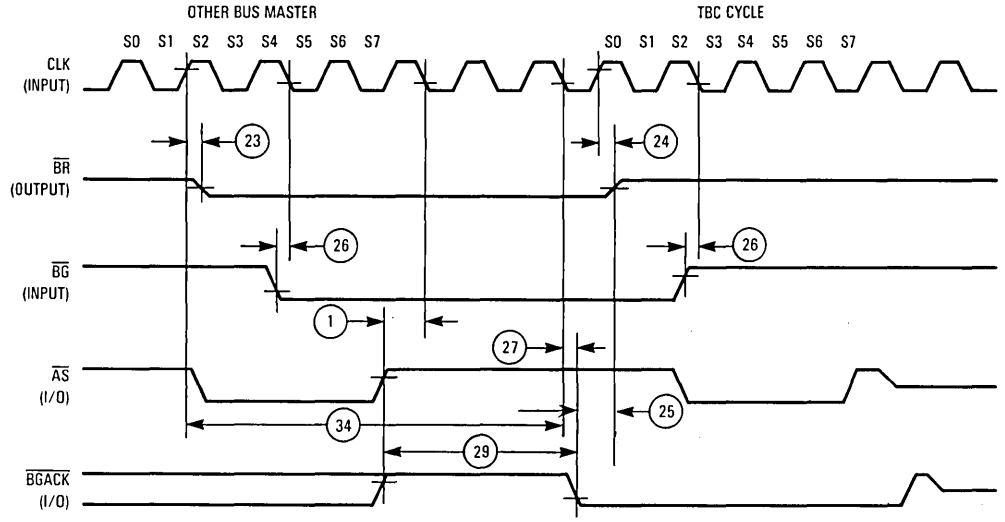


Figure 8-3. Interrupt Acknowledge Cycle



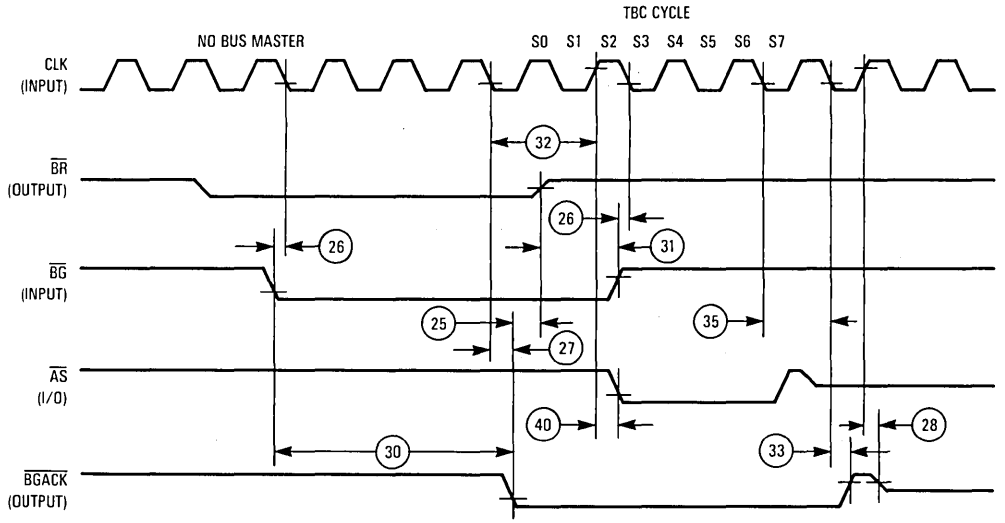
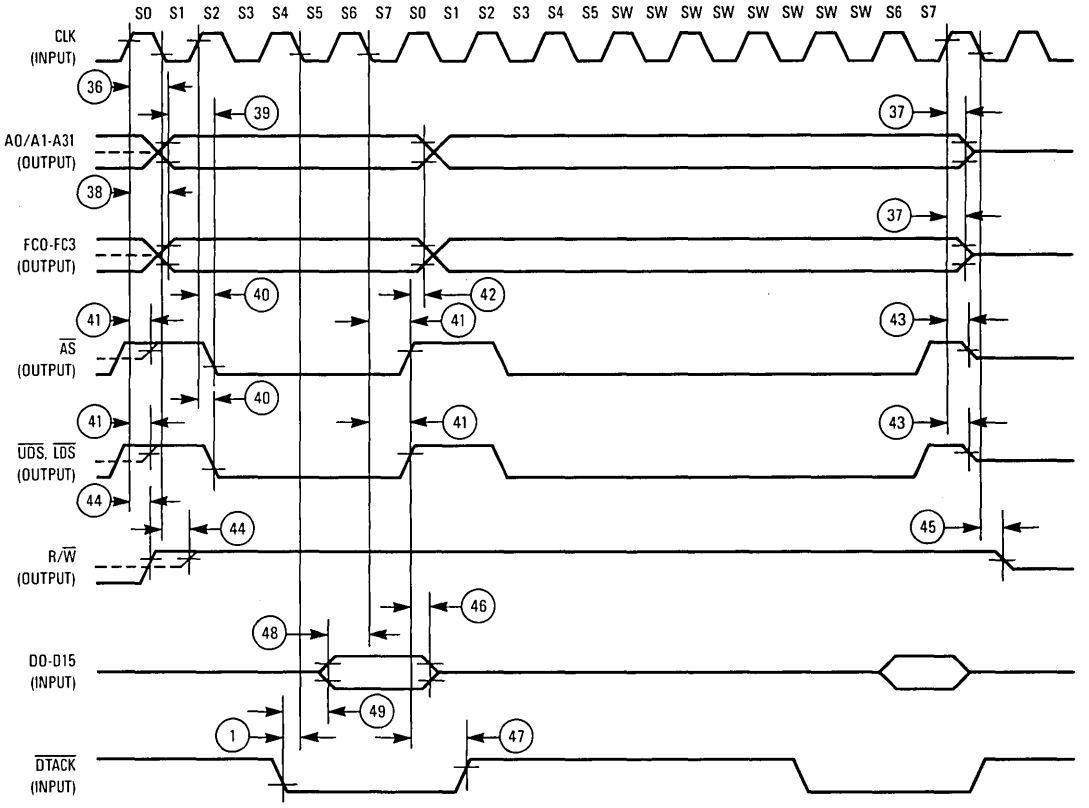


Figure 8-4. Bus Arbitration



NOTE: The solid lines assume that the communication controller was bus master on the last cycle. The dotted lines assume that there was a different bus master.

Figure 8-5. Read Cycle and Slow Read Cycle

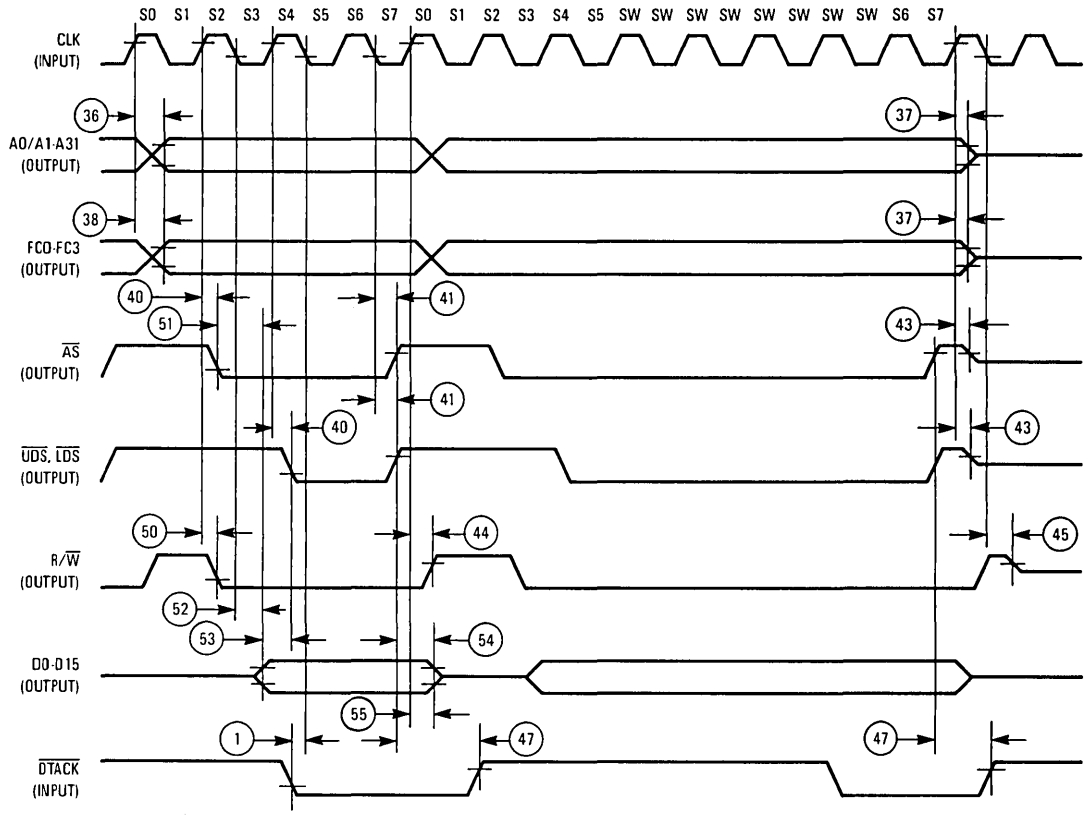
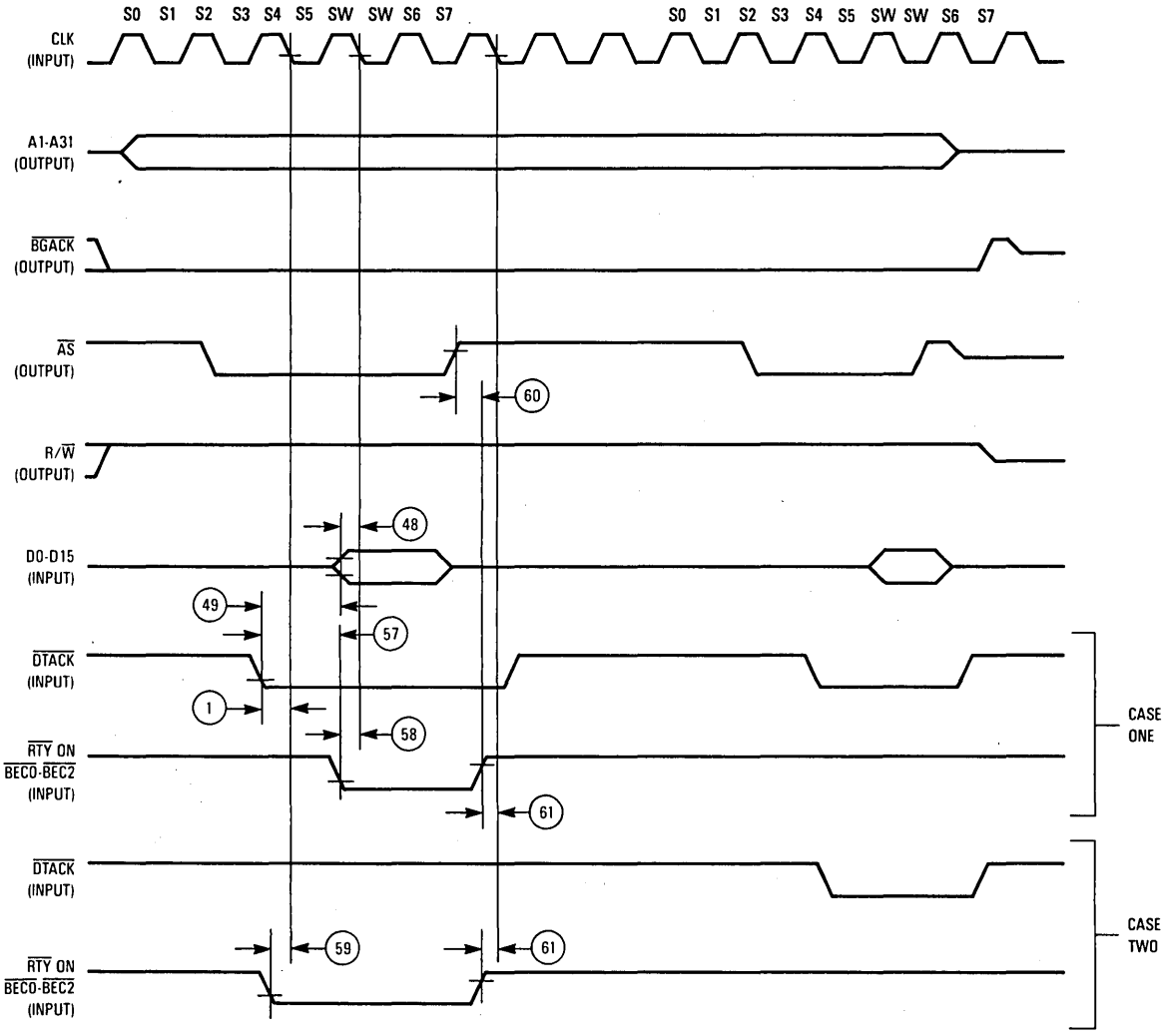


Figure 8-6. Write Cycle and Slow Write Cycle



CASE ONE: If $\overline{\text{DTACK}}$ satisfies 1 then 48 and 58 are required. If $\overline{\text{DTACK}}$ is active but does not satisfy 1 then 49 and 57 are required.

CASE TWO: If $\overline{\text{DTACK}}$ is not active then 59 is required for the exception active set-up time. Parameter 61 is always required for the exception inactive set-up time .

Figure 8-7. TBC Read Cycle with RETRY

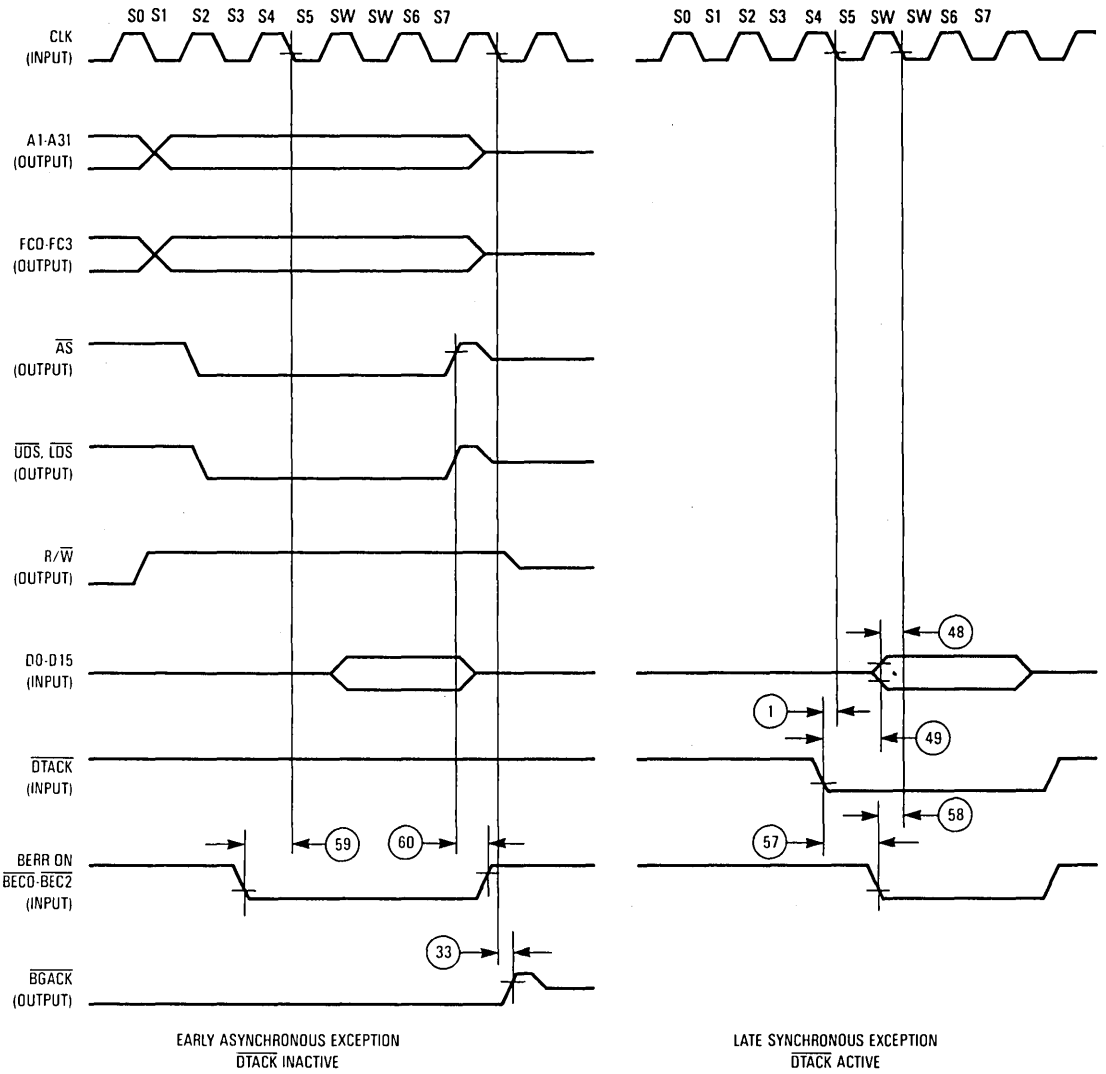
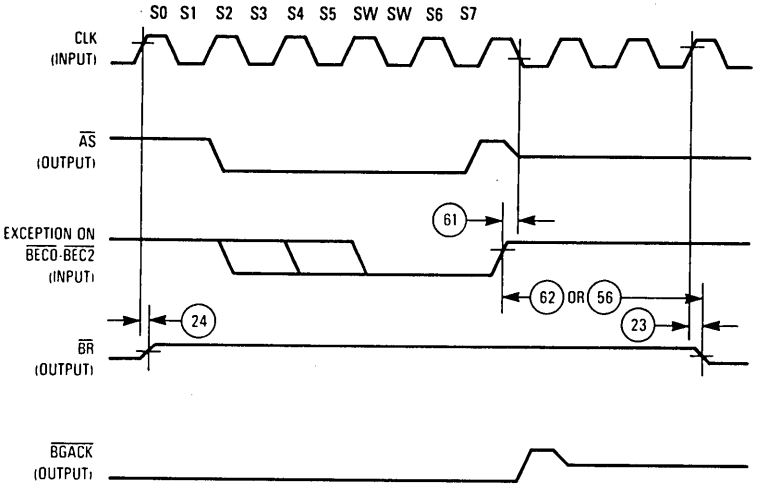
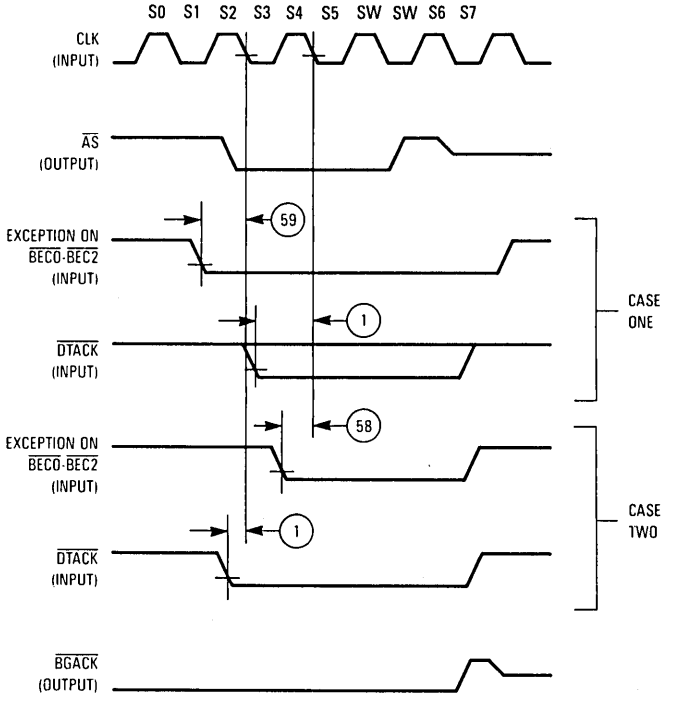


Figure 8-8. Read Cycle with Bus Error



NOTE: The above occurs when the TBC requires the bus cycle after a previous exception.

Figure 8-9. BR After Previous Exception



NOTE: Two alternatives of DTACK and exception. Case one has DTACK occur after exception and case two has exception occur after DTACK.

Figure 8-10. Short Exception Cycle

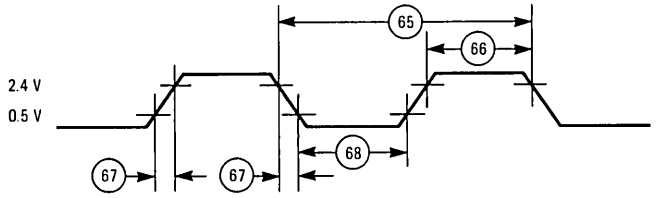


Figure 8-11. Clock, CLK

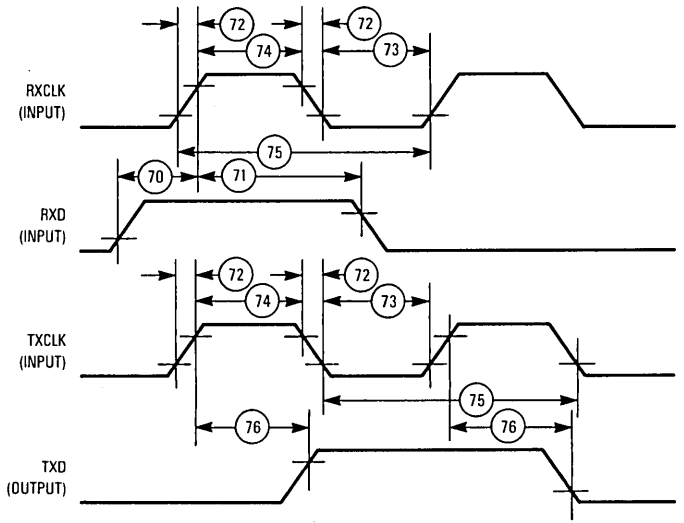


Figure 8-12. TBC Serial Data (RXD and TXD) and Serial Clocks (RCLK and TCLK)

8

SECTION 9
ORDERING INFORMATION AND MECHANICAL DATA

This section contains the pin assignments and package dimensions for the PGA (pin grid array) for the MC68824RC and for the PLCC (MC68824FN). In addition, detailed information is provided to be used as a guide when ordering.

9.1 PACKAGE TYPES

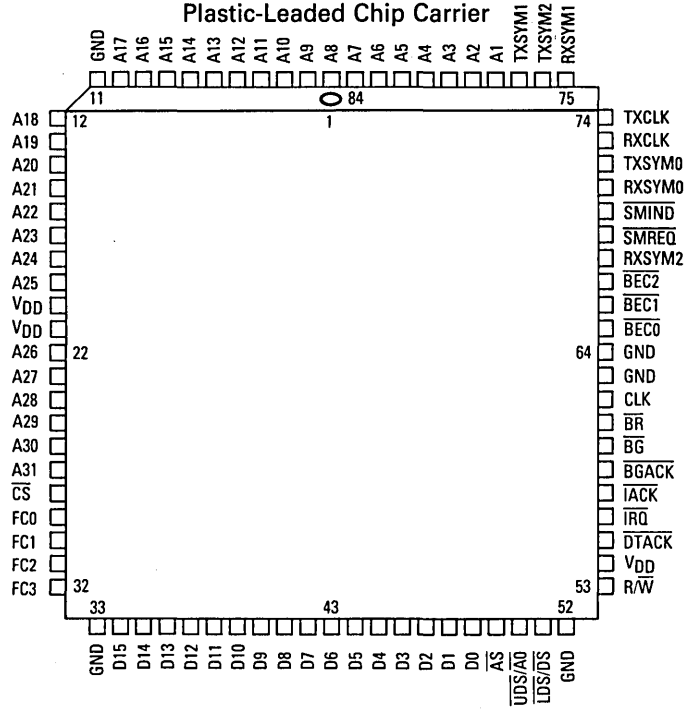
Suffix	Package Type	Comments
RC	Pin Grid Array (PGA) Ceramic	Depopulated Center Pins Gold Lead Finish No Standoffs
FN	Plastic Leaded Chip Carrier	Suitable for Socketing or Surface Mounting

9.2 STANDARD ORDERING INFORMATION

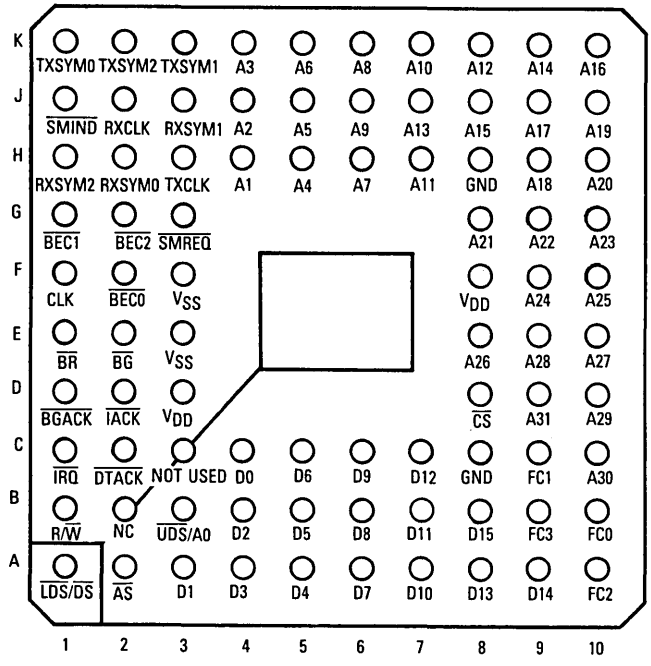
Package Type	Serial Range	System Frequency	Temperature Range	Ordering Designation
Pin Grid Array RC Suffix	1 MHz to 10 MHz	10.0 MHz	0°C to 70°C	MC68824RC10
	1 MHz to 12.5 MHz	12.5 MHz	0°C to 70°C	MC68824RC12
	5 MHz to 16.67 MHz	16.67 MHz	0°C to 70°C	MC68824RC16
	10 kHz to 10 MHz	10 MHz	0°C to 70°C	MC68824RC10S
	10 kHz to 12.5 MHz	12.5 MHz	0°C to 70°C	MC68824RC12S
Pin Grid Array IRC Suffix	1 MHz to 10 MHz	10.0 MHz	0°C to 85°C	MC68824IRC10
	1 MHz to 12.5 MHz	12.5 MHz	0°C to 85°C	MC68824IRC12
	5 MHz to 16.67 MHz	16.67 MHz	0°C to 85°C	MC68824IRC16
Plastic Lead Chip Carrier FN Suffix	1 MHz to 10 MHz	10.0 MHz	0°C to 70°C	MC68824FN10
	1 MHz to 12.5 MHz	12.5 MHz	0°C to 70°C	MC68824FN12
	10 kHz to 10 MHz	10 MHz	0°C to 70°C	MC68824FN10S
	10 kHz to 12.5 MHz	12.5 MHz	0°C to 70°C	MC68824FN12S

9.3 PIN ASSIGNMENTS

Plastic-Leaded Chip Carrier



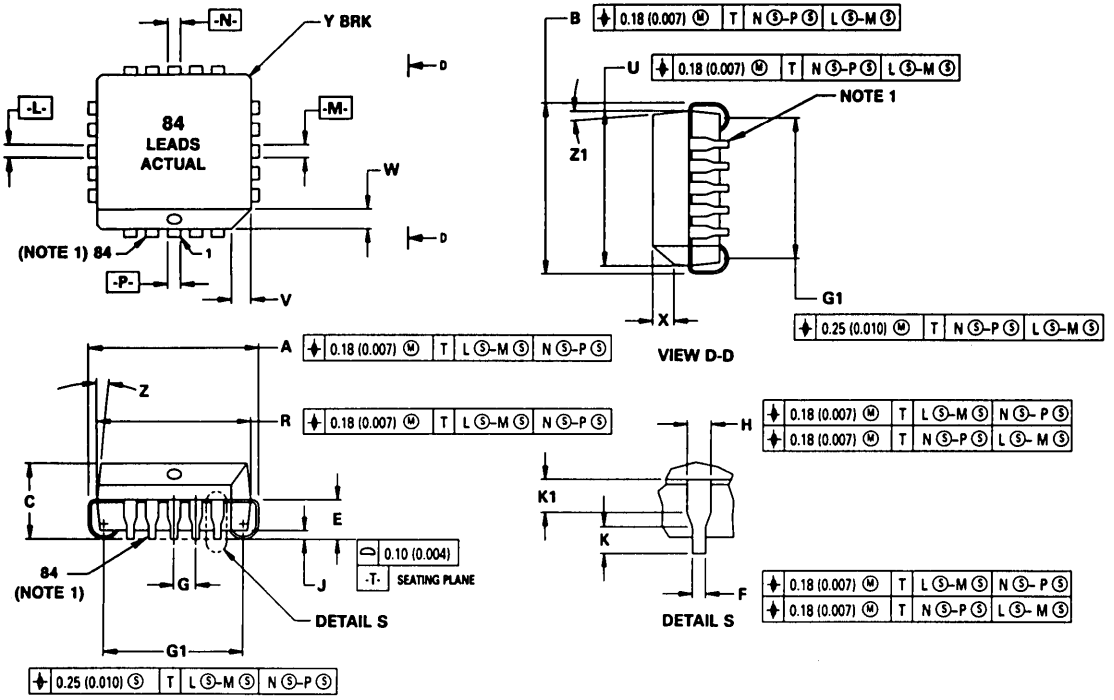
Pin-Grid Array



9

9.4 PACKAGE DIMENSIONS

FN SUFFIX
CASE 780-01
PLASTIC LEADED
CHIP CARRIER

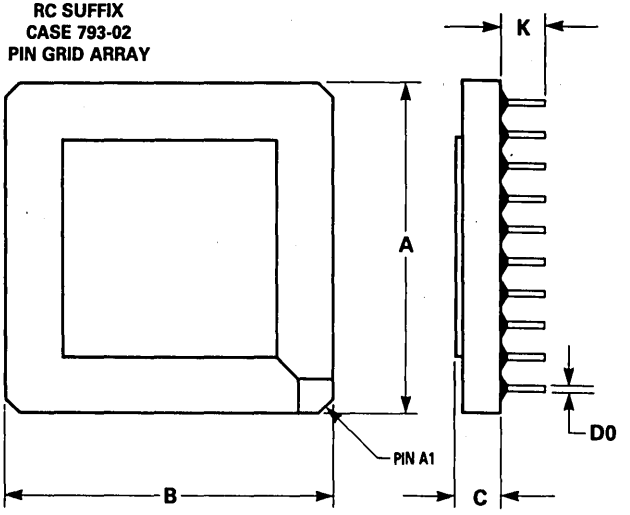


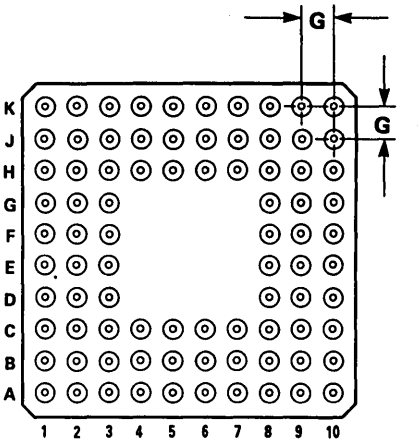
DIM	MILLIMETERS		INCHES	
	MIN	MAX	MIN	MAX
A	30.10	30.35	1.185	1.195
B	30.10	30.35	1.185	1.195
C	4.20	4.57	0.165	0.180
E	2.29	2.79	0.090	0.110
F	0.33	0.48	0.013	0.019
G	1.27 BSC		0.050 BSC	
H	0.66	0.81	0.026	0.032
J	0.51	—	0.020	—
K	0.64	—	0.025	—
R	29.21	29.36	1.150	1.156
U	29.21	29.36	1.150	1.156
V	1.07	1.21	0.042	0.048
W	1.07	1.21	0.042	0.048
X	1.07	1.42	0.042	0.056
Y	—	0.50	—	0.020
Z	2°	10°	2°	10°
G1	28.20	28.70	1.110	1.130
K1	1.02	—	0.040	—
Z1	2°	10°	2°	10°

NOTES:

1. **DUE TO SPACE LIMITATION, CASE 780-01 SHALL BE REPRESENTED BY A GENERAL (SMALLER) CASE OUTLINE DRAWING RATHER THAN SHOWING ALL 84 LEADS.**
2. DATUMS -L-, -M-, -N-, AND -P- DETERMINED WHERE TOP OF LEAD SHOULDER EXIT PLASTIC BODY AT MOLD PARTING LINE.
3. DIM G1, TRUE POSITION TO BE MEASURED AT DATUM -T-, SEATING PLANE.
4. DIM R AND U DO NOT INCLUDE MOLD PROTRUSION. ALLOWABLE MOLD PROTRUSION IS 0.25 (0.010) PER SIDE.
5. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
6. CONTROLLING DIMENSION: INCH.

RC SUFFIX
CASE 793-02
PIN GRID ARRAY





NOTES:

1. DIMENSIONS A AND B ARE DATUMS AND T IS A DATUM SURFACE.
2. POSITIONAL TOLERANCE FOR LEADS: (84 PL)

⌖	ϕ 0.13 (0.005) Ⓛ	T	A Ⓢ	B Ⓢ

3. DIMENSIONING AND TOLERANCING PER Y14.5M, 1982.
4. CONTROLLING DIMENSION: INCH.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	—	27.43	—	1.080
B	—	27.43	—	1.080
C	2.03	2.67	0.080	0.105
D	0.43	0.61	0.017	0.024
G	2.54 BSC		0.100 BSC	
K	3.56	4.95	0.140	0.195

9

APPENDIX A
IEEE 802.4 OPERATION

The IEEE 802.4 token bus standard defines a broadcast protocol where all stations on the network hear everything (or a portion of everything) that is transmitted. While the station holds the token, it has the right to transmit. All stations on the network do not have to be involved in token passing. There may be stations that just "listen" or stations that only respond to special request_with_response frames. Every station has the capability to detect and correct network error conditions such as multiple tokens or lost tokens. This means that no special "monitor" stations are required.

A.1 FUNCTIONS

The non-error condition functions of the token bus medium access control layer are to maintain steady state operation by passing the token, establish a logical ring on initialization, and allow stations to enter and leave the logical ring without disrupting the network. The error related functions are to recover from multiple tokens, lost tokens, token pass failures, non-operating receivers and transmitters, and duplicate station addresses. Explanation of error related functions are not presented here. Descriptions may be found in the IEEE 802.4 standard.

A.2 STEADY STATE OPERATION

TERMS

This Station (TS)
The address of the station that the discussion is in reference to

Previous Station (PS)
The address of the station that TS gets the token from

Next Station (NS)
The address of the station that TS passes the token to

During steady state operation, the token is passed from the highest addressed station in the network to the next highest addressed station, in descending order of address to the lowest addressed station, then back to the highest addressed station. This circular passing of the token forms a logical ring. Each station knows its previous station (PS), its next station (NS), and its own station address referred to as this station (TS). PS and NS are dynamically determined by the token bus protocol. Each station is allowed to hold the token for a user programmed amount of time. The addition of all the maximum amounts of time each station may hold the token plus the propagation delay equals the maximum token rotation time. The maximum token rotation time is a measure of the worst case time between each station's opportunity to transmit.

A.3 INITIALIZATION

TERMS

Slot Time
A measurement of time consistent throughout the network which is defined as the maximum amount of time any station need wait for an immediate response from another station. It is primarily dependent on the worst case station delay in the network. The mathematical formula is given in **2.1.24 Slot Time.**

Bus Idle (Inactivity) Timer
A timer within the Token Bus Controller MAC that determines how long the station will listen to silence before trying to initialize the logical ring. This timer is set at seven slot times unless it is the lowest addressed station when it is set to six slot times.

"Claim_Token" Control Frame
Control frame sent to initialize the logical ring. Any station that has not heard anything on the bus for a specified time (the inactivity timer expired), will send a "claim_token" frame to start the initialization sequence. A "claim_token" frame has a data field length of zero, two, four, or eight slot times, depending on the address of the station.

Initialization of the logical ring is a special case of adding new stations. One station on the network will claim the token and the algorithm is set up such that the station who claims the token is the highest addressed station. Each active station on the network monitors the medium. If nothing happens for a specified length of time, that station's bus_idle (inactivity) timer times out. When the bus_idle timer expires, the station sends a "claim_token" control frame. The length of the "claim_token" frame information field is in multiples of slot time (zero, two, four, or six) and is determined (the first time) by the two most significant bits in the station's address. Having different lengths of the "claim_token" frames is done in anticipation that more than one station will attempt to establish the logical ring at the same time. After a station sends a "claim_token" frame, the station waits one slot time for its transmission to die out, then samples the cable. If the station hears non-silence, it knows that a station with a higher address is also attempting to establish a logical ring, so the lower addressed station (TS) defers initialization to the higher addressed station. If the station heard silence, it sends another "claim_token" frame with the next two significant bits of its address determining the length of transmission. When all the address bits have been used in the above manner plus one more claim token frame whose length is determined by two random bits and silence is still sensed, the station has won the initialization sequence and now holds the token. The logical ring is built from here by adding new stations as described in the following paragraphs.

A

A.4 PASSING THE TOKEN

TERMS

"Token" Control Frame
A bit pattern used to determine when a station can transmit

"Who_Follows" Control Frame
Control frame used to determine who the station is that is the successor to this station's NS, which is contained in the data field of the frame. A "who_follows" frame is used to skip over a non-working station to delete it from the logical ring. After this frame is sent, the sending station waits three slot times for a response. An appropriate response is a "set_successor" frame.

"Set_Successor" Control Frame
Control frame which tells the requesting station who its successor should be. It is an appropriate response to any "who_follows" or "solicit_successor" control frame. Its destination address is equal to the source address of the last frame received. Its data unit is equal to the station's NS or TS.

"Solicit_Successor_2" Control Frame
Used to allow stations to enter the logical ring if it is the lowest addressed station in the network. Also, it is used to allow any station to respond when the station does not know the state of the network. Two response windows always follow this frame.

A station passes the token by encoding it in the MAC frame control field of a frame addressed to its successor. After the token is sent, the station listens to the medium to make sure the token pass has been completed successfully. If the sending station hears a valid frame, it assumes its successor has received the token correctly and is transmitting. If the sending station hears an invalid frame, it will wait and listen for up to four slot times. If anything is heard during those four slot times, the sending station assumes its successor has the token and is operating correctly. If, in those four slot times, the sending station still does not hear anything, it assumes it was the garbled token frame that it heard. The sending station will repeat the token pass once and listen to the medium again. If nothing is heard again within the four slot times, the sending station assumes its successor has failed. The sending station will now send a "who_follows" frame with its successors address in the data field of the frame. All the stations on the network compare this address to the address of their predecessor and if they match, that station will send a "set_successor" frame with its address in the data field. The sending station now knows who its new successor is and has deleted the failed station out of the ring. If the station hears no response to the "who_follows" frame, it sends a "solicit_successor_2" frame with its address as both the source address and destination address, asking any station in the network to respond. Any operational station within the logical ring should respond to this control frame. If no station responds to the "solicit_successor_2" frame, the station gives up trying to maintain the logical ring and waits for any activity on the network.

A.5 ADDING NEW STATIONS

TERMS

Any_Send_Pending
A boolean that indicates that there is something in the transmit queues to transmit for this station.

In_Ring_Desired
A boolean that indicates whether the station wishes to be a part of the logical ring, even if it has nothing to send.

"Solicit_Successor_1" Control Frame
Control frame used to allow stations between TS and NS to enter the logical ring. One response window always follows this frame.

Response Window
Measured in slot times, the response window is opened after a MAC control frame to hear a response. The parameter max_inter_solicit_count determines how often a station attempts to open a response window.

New stations attempt to join the logical ring when their any_send_pending or in_ring_desired parameters become true. New stations are added to the logical ring through a controlled contention process. On each rotation of the token for each station, the current token rotation time (measured by the ring maintenance timer) is compared to the value set by station management (max_ring_maintenance_rotation_time). If the ring maintenance timer value is less than the max_ring_maintenance_time value, that station will let a new station enter if the new station's address is between its address and the address of its successor. This occurs by the station sending out a "solicit_successor_1" control frame with its address and the address of its successor. Then the sending station waits one response window (one slot time before a station starts transmitting) for a station to begin to answer that it is within that range. Responding stations send this soliciting station a "set_successor" frame with its address in the data field so the soliciting station will make the responding station its new successor. It is possible for more than one station to respond to the "solicit_successor" at the same time. If this happens, the soliciting station sends a "resolve contention" control frame which works much like the initialization algorithm.

A.6 PRIORITY

The priority option within the IEEE 802.4 standard is implemented by the token bus controller. This option allows more important messages to be transmitted the next time the station receives the token, and less important messages to be transmitted when important messages do not need to be transmitted. There are four priority queues (access classes) for receive and four priority queues (access classes) for transmit. These access classes are six, four, two, and zero, with six being the highest access class. LLC has eight access classes defined. These eight access classes are mapped into the MAC access classes by the MAC ignoring the least significant bit.

The TBC always receives frames that are destined to it, and places them in the appropriate priority queue. The TBC only transmits out of enabled queues. The user enables the queues in the transmit status area of the private area (see **2.1.15 TX Queue Access Class Status**). Any combination of transmit queues may be enabled.

When a station receives a token and all the transmit queues are enabled, the station transmits out of transmit queue six until it either has nothing more to transmit, or the hi_priority_token_hold_time expires. If the station has finished transmitting its queue six frames or the hi_priority_token_hold_time has expired, the TBC will then check to see whether it has frames in a lower access queue to transmit. If transmit queue four has frames to transmit, the TBC checks the target rotation time for access class queue four which has been set by station management. If this target rotation time has already been reached, the station can not transmit out of this transmit queue and checks the next lower transmit queue. If this target rotation time has not yet been reached, the station will transmit out of this priority queue until either it has no more frames of this priority to send, or the target rotation time for the access class queue has been reached. Then the station checks the target rotation time for the next lower transmit queue and so on. When the lowest transmit queue is serviced, the station performs any logical ring maintenence if it is required, and then passes the token to its successor.

NOTE

This is only a summary of a portion of how the IEEE 802.4 standard works. For more information, refer to the IEEE 802.4 standard 1987. This can be purchased from:

IEEE Headquarters
345E 47th Street
New York, NY 10017-2394
Telephone: (212) 705-7960

APPENDIX B
FRAME FORMATS AND ADDRESSING

B.1 FRAME FORMATS AND ADDRESSING

The frame format used in an IEEE 802.4 network is as follows:

| Preamble | SD | FC | DA | SA | Data | FCS | ED |

Where:

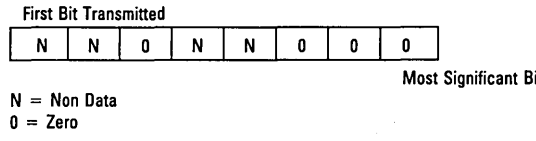
Preamble = Pattern used to set receiving modem's clock and level (one or more octets)
SD = Start Delimiter (one octet)
FC = Frame Control (one octet)
DA = Destination Address (two or six octets)
SA = Source Address (two or six octets)
Data = Information (zero or more octets)
FCS = Frame Check Sequence (four octets)
ED = End Delimiter (one octet)

B.1.1 PREAMBLE

Preamble precedes every transmitted MAC frame. It is used primarily for the receiving modem to acquire signal level and phase lock by using a known pattern. Preamble is also used to give stations a minimum amount of time to process a frame previously received. The amount of preamble transmitted depends on the data rate and modulation scheme. Preamble must always be a minimum of 2 microseconds regardless of the data rate and an integral number of octets must be sent. Therefore, on a 10 Mbit/sec network, three octets of preamble is the minimum. The maximum preamble length is constrained by the jabber control in the physical layer.

B.1.2 START DELIMITER

The start delimiter informs the modem that a frame is coming. The signaling patterns of the start delimiter are always distinguishable from data. The start deliminter is represented as follows:



B.1.3 Frame Control

The frame control field determines the type of frame. The three types of frames are MAC control, LLC control, and station management. A MAC control frame looks as follows:

First Bit Transmitted

0	0	C	C	C	C	C	C

Most Significant Bit

Where:

C	C	C	C	C	C	
0	0	0	0	0	0	claim_token
0	0	0	0	0	1	solicit_successor_1
0	0	0	0	1	0	solicit_successor_2
0	0	0	0	1	1	who_follows
0	0	0	1	0	0	resolve_contention
0	0	1	0	0	0	token
0	0	1	1	0	0	set_successor

Data frames are represented as follows:

Bits 0 1 2 3 4 5 6 7

First Bit Transmitted

F	F	M	M	M	P	P	P

Most Significant Bit

Where:

Frame Type

F	F	
0	1	LCC Data Frame
1	0	Reserved (Former Systems Management)
1	1	Reserved

MAC Action

M	M	M	
0	0	0	Request with No Response
0	0	1	Request with Response
0	1	0	Response

Priority

P	P	P		
1	1	1	Highest Priority	(6)
1	1	0		(6)
1	0	1		(4)
1	0	0		(4)
0	1	1		(2)
0	1	0		(2)
0	0	1		(0)
0	0	0	Lowest Priority	(0)

For example, an LCC data frame of priority six would have a frame control field in the TBC frame descriptor as follows:

Bits	7	6	5	4	3	2	1	0
Frame Control	0	1	1	0	0	0	1	0

B.1.4 Address Fields

Addresses may be either 16 or 48 bits in the TBC as set in bit 14 of offset 7A in the initialization table upon initialization. All addresses on a local area network must be of the same length. The least significant bit of the destination address determines if the address is individual or group (I/G bit). The I/G bit in the source address field is reserved by IEEE 802.4. The next least significant bit determines if a 48-bit address is locally administered or globally administered. Globally administered addresses are unique throughout the world and are assigned by IEEE.

B.1.5 Data

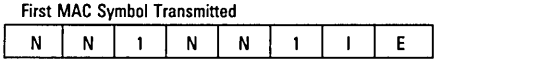
The number of octets between the start deliminiter and the end deliminiter must be 8191 or fewer, not counting SD and ED. The least significant bit is always transmitted first.

B.1.6 Frame Check Sequence (FCS)

The FCS is a 32-bit frame check sequence. The TBC always performs a frame check sequence on incoming frames. The TBC always generates an FCS for transmitted frames unless the TCDS bit is set in the SET MODE 3 command. If this mode is selected, the TBC treats the last four bytes of data as the FCS. The user may also choose to write the FCS to memory through the SET MODE 3 command. The TBC can accept frames with an incorrect FCS by setting the receive error mask in the initialization table appropriately. The term CRC (cyclic redundancy check) is used synonymously with FCS throughout this document.

B.1.7 End Delimiter

The end delimiter ends the frame and determines the position of the FCS. The end delimiter consists of signalling patterns that are always distinguishable from data. The end delimiter is represented as follows:



N = Non Data
1 = One
I = Intermediate Bit ("1" — More to Transmit, "0" — End of Transmission)
E = Error Bit ("0" — No Error, "1" — Error)

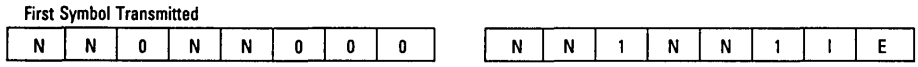
B.1.8 Invalid Frames

An invalid frame is defined by IEEE 802.4 to meet at least one of the following conditons:

1. Identified as invalid by the physical layer (contains non-data or invalid symbols).
2. Is not a integral number of octets in length.
3. Does not contain the proper fields or the fields are in an improper order.
4. The FCS computation fails.
5. The frame control field contains an undefined bit pattern.
6. The error bit is set to one.

B.1.9 Abort Sequence

An abort sequence prematurely terminates the transmission of a frame. The abort sequence is sent by a station that does not wish to continue to send a frame it has already begun sending. The TBC will automatically generate an abort sequence if the conditions require it to do so. For example, the TBC will generate an abort sequence if it gets an underrun condition while transmitting a frame. The abort sequence is a start delimiter immediately followed by an end delimiter and is represnted as follows:



B.2 ADDRESSING

IEEE 802.4 uses the IEEE addressing notation and two types of addressing: individual and group. Individual addressing identifies a particular station and group addressing identifies a group of logically related stations. These two types of addresses are differentiated by the least significant bit in the destination address field of the frame.

B.2.1 IEEE Addressing

The IEEE globally administered 48-bit address contains 12 hexadecimal digits with each digit paried in groups of two. The bits within each octet or pair, are transmitted from the least significant bit to the most significant bit. For example, an address supplied by IEEE is represented as follows:

F0-2E-15-6C-77-9B

It would be transmitted onto the local area network as follows:

First Bit Transmitted
0000 1111 0111 0100 1010 1000 0011 0110 1110 1110 1101 1001

This address would appear in the TBC private area as:

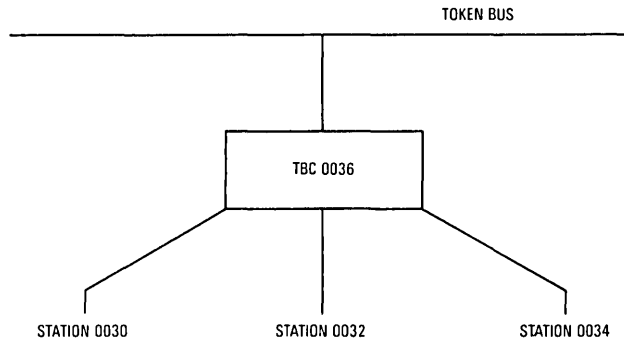
Least Significant Bit
TS Low 2EF0 TS Medium 6C15 TS High 9B77

This address would appear in the TBC frame descriptor as:

Offset 28 502E Offset 30 156C Offset 32 779B

B.2.2 Individual Addressing

The individual address, or this station (TS), of a station is set during initialization through the initialization table. The TBC also contains an individual address mask that allows the station to accept individually addressed frames for more than one station. The individual address mask is also set on initialization through the initialization table. This means the TBC can be used in the configuration below. In this example, station 0036 is the only member of the logical ring and therefore the only station to receive and send the token. Station 0036 will receive frames addressed



The IA mask is not applied to control frames and the IA mask least significant bit should always be zero. If the I/G bit is zero indicating individual addressing, the following opertion is performed to determine whether to accept the frame:

DA AND IA Mask = This Station Address (TS) AND IA Mask

Some examples are shown below. All examples have this station address (TS) equal to 0036 hex and an address length of 16 bits.

To accept only data from this station (DA = TS), the IA mask is set as below:

IA Mask = 1111 1111 1111 1110 (FFFE hex)

To accept data frames that are for one of the four stations 0030, 0032, 0034, or 0036:

IA Mask = 1111 1111 1111 1000 (FFF8 hex)

To cite a specific example, if the destination address of an incoming frame is set to 0032, the operation for acceptance of the frame is performed by the TBC as follows:

0032 (DA) ANDed with FFF8 (IA Mask) = 0036 (TS) ANDed with FFF8 (IA Mask)

Since the equation is true, the frame is accepted. A binary representation of the acceptance equation is shown below:

DA	= 0000	0000	0011	0010	
IA Mask	= 1111	1111	1111	1000	
DA AND IA Mask:					= 0030 hex
IA Mask	= 0000	0000	0011	0000	
TS	= 0000	0000	0011	0110	
IA Mask	= 1111	1111	1111	1000	
TS AND IA Mask:					= 0030 hex
TS	= 0000	0000	0011	0110	

The equation would also be true if DA = 0030, 0031, 0033, 0034, 0035, and 0036 so frames with these DAs would also be accepted.

To accept data frames that are only for one of the two stations 0036, 0136:

IA Mask = 1111 1110 1111 1110 (FEFE hex)

B

The source address identifies the station originating the frame and has the same format as the destination address, except the individual/group bit is always set to zero.

B.2.3 Group Addressing

The TBC recognizes group addressing when the I/G bit (least significant bit of the destination address) is equal to one. Only information frames can have group addresses. Messages are sent to only one group, but each station may belong to many groups. Therefore, if the sending station wants to send a message to multiple groups, a separate message is sent to each group, one for each group address.

The group address may be divided into as many fields as the user requires different groups. In the following example, four fields make up the 48-bit group address, even though not all 48 bits are used. Unused bits are set to zero. Within the example network, the following group address field definition is used:

Bit #

0	1 3	4 9	10 15	16 21	22 47
0	Floor #	Cell #	Project ID	Department ID	Zeros
0	001 Floor 1	000001 Cell 1	000001 Proj. 1	000001 Dept. 1	Zeros
0	010 Floor 2	000010 Cell 2	000010 Proj. 2	000010 Dept. 2	Zeros
0	100 Floor 3	000100 Cell 3	000100 Proj. 3	000100 Dept. 3	Zeros
etc.		001000 Cell 4	001000 Proj. 4	001000 Dept. 4	Zeros

Example of Group Address Mask

Bit #

0	1 3	4 9	10 15	16 21	22 47
0	010 Floor 2	001000 Cell 4	000011 Proj. 1 Proj. 2	000010 Dept. 2	000....00000

The equation for accepting a group address (GA) frame:

Destination Address (DA) AND GA Mask = DA

To calculate the value for the group address mask field, the logical OR of the group address is used. For example, if a station is a member of both projects 1 and 2, the station's group address mask segment for the department ID portion should be set to the logical OR of the group address field or 000011. The same is true for the rest of the fields. When assigning group addresses, zeros should be used in any field the station doesn't use. By carefully dividing the address into fields, a large number of group addresses may be supported.

A broadcast address is a group address that is all ones. A broadcast addressed frame will go to all stations on the network regardless of IA and GA masks.

B.2.4 Promiscuous Listener

The TBC can be programmed to operate in a special mode called "promiscuous listener" mode. In promiscuous listener mode, a station records everything on the network. This mode is implemented by setting the individual address mask to all zeros (receive all frames), setting the group address mask to FFF (receive all group addressed frames) and setting copy for all control frames as data (through the SET MODE command).

APPENDIX C
BRIDGING

C.1 INTERCONNECTION OF NETWORKS

Bridges, routers, and gateways are all used to interconnect networks. Bridges are the simplest of interconnecting devices and are defined to connect similar networks by using only the first two OSI layers. Routers, also referred to as "level 3 relays", are more complicated than bridges and utilize the ISO OSI level 3 function of segmenting. Gateways, also referred to as "level 7 relays", are the most complicated of interconnecting devices and connect networks with different protocol architectures. The type of interconnecting device used depends on the networks being connected and the performance required.

MAC bridges are specified by IEEE 802.1 and the MAP specification because of MAP's requirements for performance, transparency, ease of use and cost. Bridges have the best performance of any interconnecting device since they connect only similar LANs and do not require much software to operate. The end user never needs to know where a station that it addresses physically resides, meaning bridges are transparent. When nodes change location within an extended LAN, little or no human or LAN intervention is required using bridges as interconnecting devices, making them very easy to use and maintain. Since bridges have the least amount of hardware and software of any interconnecting device, they are the least expensive to implement.

Examples of where bridges are intended to be used are given in the MAP specification and include: connecting two or more identical LANs, multiplying the maximum allowable nodes and maximum allowable cable distances without affecting the token rotation times of each segment (a bridge is not a repeater); connecting LANs with different data rates (connecting 10 Mbps 802.4 broadband to 5 Mbps 802.4 carrierband); and connecting two different broadband channels on a broadband backbone. When using a bridge, it is important to consider the different frame length limitations of the segments being connected.

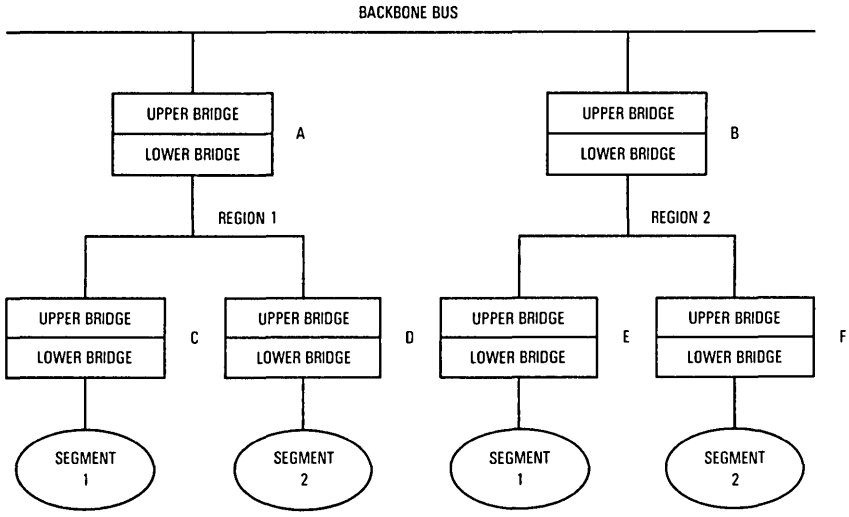
There are many different ways to implement bridging. Some of the different ways the TBC may be used to implement bridging are described in the following paragraphs.

C.2 HIERARCHICAL ADDRESSED BRIDGING

The hierarchical addressed bridging mechanism requires a hierarchy of address assignment to easily "filter" the appropriate frames into or out of segments. The discussion here focuses on the TBC MAC capabilities in hierarchical addressed bridges and how to use them. A typical interconnected network is shown below connected with hierarchical bridges.

Each hierarchical bridge is made up of two MAC entities. One MAC entity belongs to the logical ring of the network above it (upper bridge) and one MAC belongs to the logical ring of the network below it (lower bridge). The upper bridge takes frames that are destined for networks below it and passes them down to the lower bridge for transmission onto the lower network. The lower bridge takes frames that do not belong on the lower network, or on any networks below it, and

passes them up to the upper bridge to be transmitted onto the upper network. Both upper and lower bridges use the TBC individual address mask to consider only that part of the address that is relevant to it.



C.2.1 Hierarchical Addressed Bridging Implementation

Each upper bridge in the above figure is responsible for passing information down from the backbone. In the addressing scheme, the address field can be logically divided into regions. For example, the two most significant bits of the address differentiate which region the frame goes to; the next two significant bits of the address differentiate which segment the frame goes to; and the four least significant bits differentiate which station the frame goes to. For simplicity in the following example, we assume that only those eight bits form the address.

Bridges A and B recognize frames for regions 1 and 2 respectively. Bridges C, D, E, and F recognize frames for their respective segments. As an example we'll follow a frame with a destination address of 1010 0101 (region 1, segment 1, station 5).

The equation individual address (IA) AND destination address (DA) = IA AND this station (TS) is true, so bridge A accepts the frame. If the example were using a group address, the equation would be GA mask AND DA = DA.

Bridge B's individual address mask is set to C0. The address mechanism would work as follows:

IA	1100 0000	IA	1100 0000
DA	1010 0101	TS	01xx xxxx
IA AND DA	1000 0000	IA AND TS	0100 0000

The equation individual address (IA) AND destination address (DA) = IA AND this station (TS) is not true, so bridge B does not accept the frame.

Bridge C's individual address mask is set to 03. The address mechanism would work as follows:

IA	0011 0000	IA	0011 0000
DA	1010 0101	TS	0010 xxxx
IA AND DA	0010 0000	IA AND TS	0010 0000

The equation individual address (IA) AND destination address (DA) = IA AND this station (TS) is true, so bridge C accepts the frame.

C.2.2 Lower Bridges

Each lower bridge is responsible for passing frames up from the segment to the backbone. In order to be a lower bridge, the LBRM bit must be set using the SET MODE 2 command. In lower bridge mode, the bridge will forward frames only when the frame is not addressed to a station below the bridge (IA mask AND DA are not equal to IA mask AND TS or GA mask AND DA are not equal to DA). This means that lower Bridge C will forward to region 1 only those frames which do not belong in segment 1.

C.3 FLAT ADDRESSED BRIDGING OVERVIEW

Flat addressed bridges can interconnect segments where the addresses have been assigned randomly. A Spanning Tree uses flat addresses yet it is a hierarchical bridge. The Spanning Tree approach is currently being investigated by the IEEE 802.1 committee.

A table of translations is required at each station in some flat addressed bridging schemes such as IBM defined source routing. In source routing, the route taken by, or chosen for a frame as it traverses the network is reflected in the routing information field which is imbedded in the frame itself. Source routing is being proposed by the IEEE 802.5 committee for interconnection of 802.5 segments. The token bus controller provides the MAC functions necessary to implement a source routing bridge or station, making it possible to interconnect 802.4 segments into 802.5 networks or to other 802.4 segments. Stations not implementing source routing can coexist on the same segment (network) with stations that do use source routing, but cannot send messages through bridges using source routing.

If a station wishing to send a frame to another segment already knows the routing information on how to get it there, the station sends the frame with the routing information imbedded in the routing information field of the frame. If the station that wishes to send the frame does not know the routing information on how to get there, the station can dynamically discover the necessary routing information. The routing information is obtained by the station sending a "broadcast" frame which travels over every bridge in the network, eventually reaching its destination. On their way, broadcast frames record where they have been. If a bridge receives a broadcast frame which has already been on the segment it connects to (it finds its identifier in the routing information

field), it does not retransmit that frame. When the destination station receives the frame, it responds using the route noted in the frame. Multiple frames may reach the destination station because of different routes that exist, and each of these multiple frames received will be sent back to the originating station by the route that it took to get to the target station. The sending station will choose which route to save in its routing table according to different rules. Examples of these rules include: the frame that was received first, or the frame that took the shortest path (went through the least number of bridges).

There also exists a limited broadcast mode, chosen by the SRLB bit in the SET MODE 2 command, in which only those stations that are in limited broadcast mode may rebroadcast a broadcast frame. This reduces the number of frames the receiving station gets by strategically placing the limited broadcast stations, thus reducing the amount of traffic on the network.

C.3.1 Source Routing Implementation

The source routing mechanism is implemented when the source address (SA) I/G bit is equal to one. In the IEEE 802.4 standard, the SA I/G bit equal to one is reserved for future use. When using source routing, the routing field is a subset of the data field of the 802.4 standard. The frame format is shown as follows:

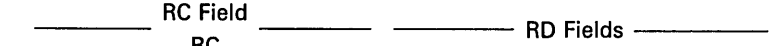
NORMAL 802.4 FRAME

SD	FC	DA	SA	DATA	FCS	ED

SOURCE ROUTING IMPLEMENTED

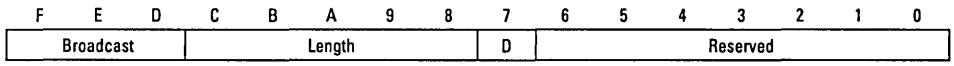
SD	FC	DA	SA IG=1	RI	DATA	FCS	ED

The routing information (RI) field is present only when the SA field least significant bit, referred to as the routing information indicator, or RII bit, is set to one. The RI field can be from 2 to 30 octets. If the RII bit is equal to zero, a routing information field is not present. If this bit is one in a control frame, the frame is considered invalid and is treated as a noise burst. If this bit is one in a data frame and recognize source routing (RSR) is not enabled in the TBC, the TBC treats the frame as a normal frame. If this bit is one and RSR is enabled, the frame is accepted if the destination address matches the specified criteria set by the host or if the RI field specifies that the frame is to be routed to another segment via the TBC. In this case, the RI field is copied to the first data buffer like normal data. Note that lower bridge mode and recognize source routing mode are mutually exclusive modes. If both mode bits are set, RSR dominates. The routing information field has the format shown below.



Where:
RC The Routing Control (2 Octets)
RD Route Designators (2 Octets Each)

The format of the routing control octets is shown below:



Broadcast

The coding is as follows:

0XX — Non-Broadcast
10X — All Routes Brodcast
11X — Limited Broadcast

Where an X indicates don't care.

Non-broadcast frames always contain the specific route through the network which the frame will take. All routes broadcast frames are transmitted on every route to the destination address. Limited broadcast frames are only retransmitted by bridges who are in the special limited broadcast mode. This restricts the number of routes a broadcast frame will take, thus reducing the traffic on the network.

Length

The length of the RI field, including the control field and the route designator field is measured in octets. The length field consists of five bits.

D — Direction Bit

The direction bit indicates to a bridge whether a frame is travelling from the originating station to the target or vice versa. This bit allows the route designators to appear in the same order regardless of the direction of transmission.

Reserved Bits

Reserved bits are set to zero when transmitting, and ignored when receiving.

When the recognize source routing (RSR) bit is set by the SET MODE 2 command, the TBC provides source routing based frame reception in addition to normal address based frame reception. The TBC does not by itself provide a bridge function but does receive and transmit frames containing the source routing fields as any other MAC frames. Frames are copied based on recognition of a segment pair as described in route designators. To implement a bridge, a MAC function (such as the TBC) for each segment the bridge is attached to is required, as well as a host to provide the additional bridge functionality.

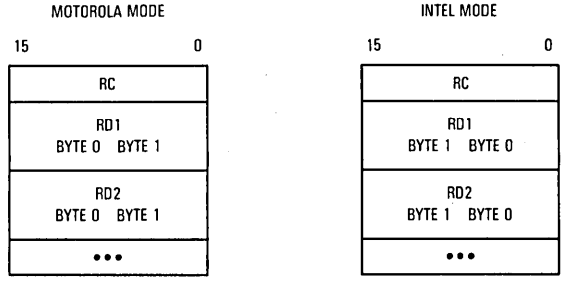
C.3.2 Route Designators

Route designators (RDs) are pairs of octets which indicate the specific segments and bridges through which the frame passes. Route designators contain bit strings with no arithmetic significance; i.e., concepts like addition, subtraction, greater than, or less than, do not apply to route designators. Thus, the two bytes of an RD can not really be described as high and low, or most significant and least significant. What is preserved in all 802 LANs is the byte transmission order. The bit transmission order of each octet depends on the type of MAC. IEEE 802.4 transmits the least significant bit first.

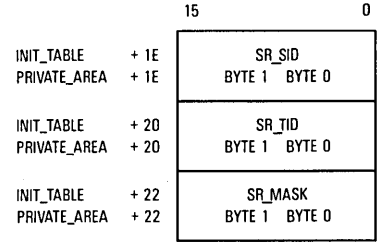
C

Each route designator can be shown as follows:

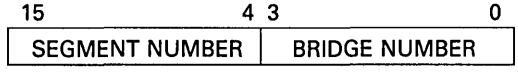
The first byte to be transmitted is byte 0 and the second byte to be transmitted is byte 1. The TBC data buffer looks as follows:



In order to perform source routing based reception, the TBC requires three parameters: SID, TID, and SR_MASK. The initial values of these parameters are taken from the initialization table. They may be changed during operation by the SET VALUE command. They have the following format:



As proposed by IEEE 802.5, each RD has the following format:



The segment number identifies a specific segment. The bridge number identifies a specific bridge between that segment and another segment. Thus, two segments connected by three different bridges would have three different bridge numbers creating three distinct routing designators. Bridge numbers are local to segment pairs, meaning bridge numbers can be the same as long as they are not connecting the same two segments.

Source ID (SID) is the identification of the segment where the frame comes from. The target ID (TID) is the identification of the segment where the frame is going to. The source routing mask (SR MASK) is a two-octet parameter defining which bits in the RDs are the segment number (denoted by a "1") and which bits are the bridge number (denoted by a "0"). The definition of the RD currently calls for 12 bits for the segment number and four bits for the bridge number, or a mask of F0FF. For broadcast frames with the RI bit set, the TBC will scan the RII field of ordered RD pairs and accept frames based on the value of SR SID and SR TID. On non-broadcast frames, accept/reject criteria are based on the values of SR SID and SR TID.

C

C.3.3 Source Routing Operation

When a station wants to send a frame to another station on a different LAN, it first sends a broadcast frame to all segments. (The host must prepare the frame, the TBC transmits it). In this frame, the routing information indicator (RII bit) is set to one, the broadcast field is set to 1XX, and the direction bit is set to zero. The source address is the individual address of the transmitter,

and the destination address is the individual address of the target station. This frame goes out from all bridges on the originating LAN. (The bridge TBC will receive the frame, the host must either add the information to the RC field and transmit the frame onto the next segment or discard it). Each bridge scans the RD field for the number of the target segment it attaches to and, if it is there, the bridge will not forward the frame because it has already been on that segment. If the bridge's next segment number is not there, the bridge adds its segment and bridge numbers to the frame's RI field, increments the RI length by two, and forwards the frame. A number of differently routed frames may arrive at the target station. The routing designator is copied for each bridge as the frame travels through in the network. (The host must add this information before transmitting the frame onto the connecting segment). The first routing designator is the RD of the first bridge to copy the broadcast frame. The target station responds to each differently routed frame by sending it back to the transmitting station. (The target host must change the RC field and put the frame into the transmit queue). Each follows the route of its routing designator field in the opposite direction. In these frames that are sent back, the RII bit is set to one, the RI field is as it arrived, the broadcast field is set to 0XX, the D bit equals one, the SA is the individual address of the target, and the DA is the individual address of the station that sent the original broadcast frame. The sending station receives as many responses as there are routes. The host at the sending station chooses the route according to its specific criteria, then saves this route and uses it for all subsequent communications with that DA. The target learns of this selected route when it receives its first non-broadcast frame.

The host is responsible for changing the routing control field if necessary. The TBC does not interpret the routing control field. The TBC transmits a source routing frame exactly as it appears in its transmit queue (i.e., the host must prepare the frame according to the source routing protocol).

More detail can be found on bridging in: IEEE 802.5, IEEE 802.1, and MAP.

C

APPENDIX D
PERFORMANCE

It is necessary to understand certain aspects of the token bus controller to compute certain network parameters such as slot time. The following describes simulations that were done to give the user the required performance numbers. In general, the numbers given are for the worst case situation. The simulations were run for network data rates of 10 Mbps and 5 Mbps. The numbers given are in symbol times (bits) elapsed from the end of the end delimiter (ED) to the start of TBC preamble and take into account the I/O sampling delays in the TBC.

System Parameters:

Address Length: 48 bits

TBC mode bits are in their default state except predefined response mode is enabled, in_ring desired is enabled, and all statistics are enabled.

Bus Width: 16 bits

Bus Latency: one cycle

Wait States: none

Commands: none

BD and FD pools are not empty with no warning bits

Preamble: minimum length

Interrupt Status Bits: none are set

Counters: not equal to thresholds

For performance reasons, it is required that a frequency ratio of systems clock to serial clock is greater than 1:1.

DESCRIPTION OF CASES

0. The TBC receives a token.
1. The TBC observes a three byte, group addressed data frame not for the TBC, and immediately after, receives a token frame that is addressed to the TBC.
2. The TBC receives a three byte data frame that is a broadcast address, and immediately after, receives a token frame that is addressed to the TBC.
3. The TBC receives a three byte, group addressed data frame not for the TBC, and immediately after, receives a solicit_successor_1 control frame that it must respond to.
4. The TBC receives a three byte data frame that is a broadcast address, and immediately after, receives a solicit_successor_1 control frame that it must respond to.
5. The TBC receives a three byte request with response frame that it must respond to (remember the TBC is in predefined response mode).
6. The TBC receives a token frame addressed to it right after the TBC has passed the token (two station ring).
7. The TBC observes a three byte, group addressed data frame not for the TBC, and immediately after, receives a token frame that is addressed to the TBC, just after it has passed the token (two station ring).
8. The TBC receives a three byte data frame that is a broadcast address, and immediately after, receives a token frame that is addressed to the TBC just after it has passed the token (two station ring).
9. The TBC receives a three byte request with response frame (predefined response mode) for the TBC, just after the TBC has passed the token.
10. The TBC receives a set_successor control frame for itself, immediately following this it receives the token, just after it has passed the token (two station ring with no_successor_1).

TBC CONTRIBUTION TO STATION DELAY
2A60S/1B59B MASK SETS

Case	systems clock:serial clock in MHz					
	12.5:10	12.5:5	10:10	10:5	8:10	8:5
0	66	35	105	42	153	53
1	66	35	150	42	261	53
2	310	38	460	94	647	184
3	78	40	121	48	173	59
4	322	43	476	102	667	194
5	376	156	498	199	651	268
6	124	66	179	81	246	100
7	135	35	242	41	375	52
8	382	68	551	139	761	242
9	449	186	589	242	765	326
10	133	35	249	42	390	79

D

MC68824

This first edition of the Motorola MC68824 Token Bus Controller User's Manual provides the latest, complete information to engineers using the MC68824. This manual includes programming information as well as detailed signal descriptions and electrical specifications. Also provided is ordering information and mechanical data to aid the user in selecting the best part for his application.

The Motorola MC68824 Token Bus Controller is a silicon integrated circuit which implements the media access control (MAC) portion of the IEEE 802.4 standard. This standard has been selected for the Manufacturing Automation Protocol (MAP) specification. The MC68824 built in features make it ideally suited for many communications applications.