XFA Port Specification

Revision 4.3

This document specifies the port interface of the XFA, the XMI to FDDI Adapter.

Issued by: VAX Products and Options

George Nielsen - Product Manager

Gerard Koeckhoven - Engineering Product Manager

Satish Rege - System Architect

Martin Griesmer - DEXFA Firmware Project Leader

Michael Nohelty - DEXFA Diagnostic Project Leader

Andy Russo - DEXFA Hardware Engineering

Ron Edgar - DEXFA Hardware Engineering

Dave Gagne - VMS Device Driver

Dave Johnson - CSSE

CONTENTS

FIGURES

TABLES

Preface

This document constitutes the port specification for the XFA (XMI to FDDI adapter). It defines the interface and provides specific implementation details.

Associated Documents

- DIGITAL: DEMNA Port Specification, Revision 4.0, Jan-1989.
- DIGITAL: Calypso Memory Interconnect (XMI), Revision 1.5.
- DIGITAL Network Architecture Maintenance Operations Functional Specification, XXX4.0.1, 29-Sep-1989.
- IEEE STD 802.2 Logical Link Control, 1985.
- DIGITAL: Parser Sub-System Design Specification, Rev 1.2, Feb 1990.
- DIGITAL: XFA Module Built-In Diagnostic Functional Specification, Rev 1.0, June 30, 1989.
- DIGITAL: XFA Hardware Functional Specification, Version 1.1, Jan, 1991.

Table 1: Revision History:

Date	Revision	Summary of Changes
14-Sep-1988	0.11	First cut-and-paste of Chapters 1 and 2 of the XFA Port Specification
1-Dec-1988	0.12	Redo of Chapters 1, 2, 3 and 4 of the XFA Port Specification
10-Feb-1989	0.13	Rewrite of the XFA Port Specification
20-Mar-1989	0.14	Internal Review of Document
28-Mar-1989	1.0	External Review of Document
25-Apr-1989	2.0	Corrected Version of Document
4-Aug-1989	3.0	Updated Version of Document
20-Nov-1989	3.1	Updated Version of Document
20-Feb-1990	4.0	Updated latest changes and added SET SMT command
5-Mar-1991	4.1	Updated for Group Review
1-Nov-1992	4.2	Updated to include CNS 5.0 changes
11-Dec-1992	4.3	Minor changes for FDX

CHAPTER 1

INTRODUCTION

The XFA is a high-performance XMI to FDDI adapter. It is being developed by VAX Products and Options in Littleton as a sub-contract from the NAC (Networks and Communication) organization in TAY and is part of a set of DEC FDDI product offerings. The adapter implements a single FDDI attachment.

This document specifies the port interface and some implementation details of the XFA.

1.1 Port Interface

The XFA Port Interface uses four rings located in host memory. These four rings are used to provide a communications protocol between the XFA port driver and the XFA adapter. The first two rings, TRANSMIT and RECEIVE are for frames transmitted to and received from the FDDI network. The third, COMMAND, is for Commands and Responses from the host to the Adapter, not destined for the FDDI network. The fourth ring, UNSOLICITED, is for host-directed unsolicited event information from the adapter to the host. The use of four rings isolates the time-critical host processor-FDDI communications, the RECEIVE and TRANSMIT rings, from the less critical port driver to adapter path, the COMMAND and UNSOLICITED rings.

1.2 Terminology and Conventions

1.2.1 Terminology

Some terms used in this specification are defined here.

- 802. 802 refers to the class of local area networks defined in IEEE 802 specifications. In this document it refers specifically to the IEEE 802.2 specification (data link layer).
- 802 SAP (Service Access Point). An FDDI frame using 802.2 data link layer contains both an DSAP and a SSAP. The source SAP (SSAP) identifies the user sending the frame. The destination SAP (DSAP) identifies the user to receive the frame.
- 802 SNAP (sometimes called Extended 802). In the case where a frame's destination SAP and source SAP consist of alternating ones and zeros, AA (HEX), and the Control Field is UI, this is called the SNAP SAP (Subnet Access Protocol SAP). This indicates that an additional five byte protocol identifier (PI) field will follow the control field. The five byte PI field identifies the particular user.
- Adapter. The hardware and firmware entity which comprises the XFA module functionality. It provides an interface, via the XMI bus, between the Host System Port Driver/Memory resident data structures and the FDDI network.
- Adapter Manager. Used interchangeably with 'firmware'. Refers to 68020 subsystem and firmware.

- Buffer Segment. A physically contiguous portion of a Buffer.
- CNS (Common Node Software). Software developed for all 3 FDDI products which control the operation of the FDDI chipset and hence the operation of the FDDI ring.
- Command. A Command Buffer for the adapter placed on Command Ring by the host.
- Command Ring. A ring located in host memory. After port initialization it has a fixed size and location. Command Ring entries contain pointers to Command Buffers. Command buffers contain commands intended for the adapter to process and exclude direct commands to send a frame on the FDDI ring. Commands which require responses, such as requests for statistics, use pre-assigned locations in the command buffer to fill out the required data. On completion, command buffers are passed back to the host.
- FDDI. FDDI (Fiber Distributed Data Interface) refers to the suite of ANSI Specifications (PMD, PHY, MAC, SMT) listed above. In this document an FDDI frame (from a adapter point of view) contains a Frame Control (FC) byte, a 48 bit Destination Address (DA), a 48 bit Source Address (SA), zero to 4478 information bytes and 4 bytes of CRC (FCS).
- Frame. An FDDI frame.
- Host. The host is the intelligence which appears to the adapter as a single processor directing its operation. The port driver software runs on the host. From the viewpoint of the adapter, the host and the port driver appear to be the same entity and the terms are sometimes used interchangeably.
- Host Interrupt. The adapter generates interrupts to the host which are fielded and handled by the port driver. These interrupts are generated when a ring entry ownership changes from adapter to host and the host is not currently servicing ring entries. Host interrupts are enabled when the port driver initializes the adapter and supplies the adapter with the necessary information to generate an interrupt.
- "My Long Address", the 48-bit FDDI station address stored in Default Physical Address ROM on the adapter. This is also referred to as the Default Physical Address.
- Port. The entity which implements the port interface on the adapter side. It performs actions requested of it by the port driver.
- Port Data Block. The data structure used during port initialization to give the adapter knowledge of the location of data structures in host memory to be used by the adapter and port driver to exchange data. It remains in existence after initialization as a place to keep counters and error information.
- Port Driver. The software running on the host processor which directs the operation of the port. It processes command, transmit, and receive requests from users of the operating system, and processes adapter unsolicited events.
- Port Interface. This is the definition of the communications protocol and associated data structures between the port driver and the adapter.
- Port Interrupt. The port driver does not interrupt the port directly. Instead, the port driver signals the port whenever it writes a port register. The port will recognize port driver register writes and service them.
- Receive Ring. A ring located in host memory. After port initialization it has a fixed size and location. Receive Ring entries contain pointers to Receive Buffers. Receive Buffers are used by the adapter as storage for frames received from the FDDI network and intended for Host Delivery.

- Ring. A host-resident data structure used to coordinate communications between the port driver and the adapter. After initialization, the port driver and the adapter traverse the rings processing (i.e. reading and writing) commands, responses, transmit requests and received frames.
- RTOS. Real Time Operating System. 68020 operating system running in the adapter.
- State Command. A write by the host to either the XPCI, XPCS or XBER (halt,reset) adapter registers.
- Transmit Ring. A ring located in host memory. After port initialization it has a fixed size and location. Transmit Ring entries contain pointers to Transmit Buffers. Transmit Buffers are intended for data which the Adapter transmits in a frame on the FDDI network.
- Unsolicited Ring. A ring located in host memory. After port initialization it has a fixed size and location. The Unsolicited Ring is used as a path for adapter-initiated communication to the host. Each entry is one octaword in length and is used directly by the Adapter Manager to pass unsolicited information to the host.

Figure 1–1: FDDI/802.2 FRAME FORMAT

```
  1    6    6
+----+----+----+----------------------------------------+---+
| FC | DA | SA |                FDDI  INFO              |FCS|
+----+----+----+----------------------------------------+---+
               .                                        .
               .   802.2 Format -                       .
               .   1      1        1          <4573     .
               +------+------+---------+----------------+
               | DSAP | SSAP | CONTROL |    USER  INFO  |
               +------+------+---------+----------------+
               .                                        .
               .   Extended 802.2 Format -              .
               .   1      1       1      5     <4568    .
               +------+------+---------+----+-----------+
               |  ^AA |  ^AA |   UI    | PI | USER INFO |
               +------+------+---------+----+-----------+
               .                                        .
               .   Extended 802.2 with Mapped           .
               .   Ethernet Frame -                     .
               .   1      1       1      5              .
               +------+------+---------+----+-----------+
               |  ^AA |  ^AA |   UI    | PI | USER INFO |
               +------+------+---------+----+-----------+

               PI format for Encapsulated Ethernet
               contains the Ethernet Type field in
               the low order two bytes.  The  high
               order 3 bytes = 0.
```

1.2.2 Abbreviations

Commonly used abbreviations are given here for reference purposes:

- CNS - Common Node Software
- LLC - Logical Link Control.
- MAC - Media Access Control.
- MOP - Maintenance Operations Protocol.
- PDB - Port Data Block (contains port initialization data).
- RO - Read Only.
- R/W - Readable and Writeable.
- R/W1C - Readable and Write '1' to Clear.
- RTOS - Real Time Operating System
- W1C - Write '1' to Clear.
- XFA - XMI to FDDI adapter.

Some IEEE 802 and ANSI abbreviations are listed here. For more information on these, refer to the DNA, ANSI and 802 specifications listed in References.

- MLA - "My Long Address", the 48-bit FDDI station identifier stored in Address ROM on the adapter.
- SAP - Service Access Point.
- DSAP - Destination SAP.
- SSAP - Source SAP.
- GSAP - Group SAP, an SAP associated with multiple users.
- SNAP - Subnet access protocol.
- SNAP SAP - Special type of SAP with DSAP and SSAP set to a value of AA (hex) and CNTL=UI, used with extended 802 frames.
- PI - Protocol Identifier, given with the SNAP SAP for extended 802 frames.

1.2.3 Other Conventions

All numbers in fields, offsets, and bit positions are given in decimal, unless explicitly stated otherwise.

MBZ (Must Be Zero) fields must contain zero on delivery of the command or specification of the data structure.

RESERVED (RSVD) fields must be zero on delivery of the command or specification of the data structure. RESERVED fields are ignored by the port. These fields can be defined to take on new meaning in future implementations without effecting previous implementations.

RESERVED (to port driver) fields are reserved for port driver usage. These fields are not changed by the port.

RESERVED (to port) fields are reserved for adapter usage. These fields should not be changed by the port driver.

NIO (to port) fields are not implemented, zero.

CHAPTER 2

PORT COMPONENTS

The XFA Port Specification is a hardware/software specification of the protocol between the XFA port driver and the XFA adapter. It consists of a set of port components which are used to provide certain functions. These components are:

...in the Adapter

- XMI Required Registers
- XFA Specific Registers

...in Host Memory

- Port Data Block
- Command, Transmit, Receive and Unsolicited Rings
- Data Buffers

2.1 XMI Required Registers

There are 4 XMI registers:

- XMI Device Register (XDEV)
- XMI Bus Error Register (XBE)
- XMI Failing Address Register (XFADR)
- XMI Failing Address Extension Register (XFAER)

The XMI registers are briefly described here. A more complete description of them can be found in the XMI specification.

2.1.1 XMI Device Register (XDEV)

The XMI Device Register contains information to identify the adapter on the XMI. It contains zero after power up or node reset. It is loaded with the device type and revision level at start of adapter self test. (Adapter written, host read)

```
 31    ...     24 23    ...    16 15          ...          0
+---------------+--------------+--------------------------+
|         Device Revision      |        Device Type       |
|   Hardware    |   Firmware   |                          |
+---------------+--------------+--------------------------+
```

Where:

- Device Type. This is the device identifier assigned to the XFA. The value is 0823 (hex), which identifies the adapter as the XFA.
- Device Revision, firmware. This is a firmware/diagnostic revision level.
- Device Revision, hardware. This is the hardware revision level numbered 0..255 mapped directly to alphabetic revision codes A-Z, AA-AZ, BB-BZ, and so forth. Some of the revision codes are not used (they are disallowed), such as G, I, O, Q, X, so the numeric equivalents will be skipped.

The contents of the XDEV register for the first 10 revisions of the module, with firmware revision XX, will be as follows:

1. 00XX0823 - Module revision A
2. 01XX0823 - Module revision B
3. 02XX0823 - Module revision C
4. 03XX0823 - Module revision D
5. 04XX0823 - Module revision E
6. 05XX0823 - Module revision F
7. 07XX0823 - Module revision H (G was skipped)
8. 09XX0823 - Module revision J (I was skipped)
9. 0AXX0823 - Module revision K
10. 0BXX0823 - Module revision L

The contents of the XDEV register for module revision C and four firmware revisions would be shown as follows:

1. 02000823 - Module revision C
2. 02010823 - Module revision C-1
3. 02020823 - Module revision C-2
4. 02030823 - Module revision C-3

2.1.2 XMI Bus Error Register (XBE)

This register is fully defined in the XMI specification but it is given here to define some of the bits that are referred to later.

A complete description of the read/write characteristics of are defined in the XMI specification.

```
 3 3 2 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1
 1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|0|0|0|1|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|0|1|1|    FCID   |0|0|0|0|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 | | | | | | | | | | | | | | | | | | | | | | | __________| | | |
 | | | | | | | | | | | | | | | | | | | | | | | | _________| | |
 | | | | | | | | | | | | | | | | | | | | | | | | | _________| + RESERVED
 | | | | | | | | | | | | | | | | | | | | | | | | | |
 | | | | | | | | | | | | | | | | | | | | | | | | | + Enable MORE Protocol
 | | | | | | | | | | | | | | | | | | | | | | | | |     (EMP)
 | | | | | | | | | | | | | | | | | | | | | | | | + Disable XMI Timeout (DXTO)
 | | | | | | | | | | | | | | | | | | | | | | | + Enable HexaWord Write (EHWW)
 | | | | | | | | | | | | | | | | | | | | | | + Failing Commander ID (FCID)
 | | | | | | | | | | | | | | | | | | | | | + Self-test Fail (STF)
 | | | | | | | | | | | | | | | | | | | | + Extended Test Fail (ETF)
 | | | | | | | | | | | | | | | | | | | + Node-Specific Error Summary (NSES)
 | | | | | | | | | | | | | | | | | | |
 | | | | | | | | | | | | | | | | | | |
 | | | | | | | | | | | | | | | | | | |           Commander Errors
 | | | | | | | | | | | | | | | | | | |   ---------------------------
 | | | | | | | | | | | | | | | | | | + Transaction Timeout (TTO)
 | | | | | | | | | | | | | | | | | + RESERVED
 | | | | | | | | | | | | | | | | + Command NoAck (CNAK)
 | | | | | | | | | | | | | | | + Read Error Response (RER)
 | | | | | | | | | | | | | | + Read Sequence Error (RSE)
 | | | | | | | | | | | | | + No Read Response (NRR)
 | | | | | | | | | | | | + Corrected Read Data (CRD)
 | | | | | | | | | | | + Write Data NoAck (WDNAK)
 | | | | | | | | | | |
 | | | | | | | | | | |        Responder Errors
 | | | | | | | | | | |    ----------------------
 | | | | | | | | | | + Read/IDENT Data NOACK (RIDNAK)
 | | | | | | | | | + Write Sequence Error (WSE)
 | | | | | | | | + Parity Error (PE)
 | | | | | | | + Inconsistent Parity (IPE)
 | | | | | | |
 | | | | | | |         Miscellaneous
 | | | | | | |   ------------------------
 | | | | | | + Write Error Interrupt (WEI)
 | | | | | + XMI Fault (XFAULT)
 | | | | + Corrected Confirmation (CC)
 | | | + XMI Bad (XBAD)
 | | + Node HALT (NHALT)
 | + Node Reset (NRST)
 + Error Summary (ES)
```

Note: On power up or node reset all bits are clear except for Self-Test Fail (STF), XBAD, and ETF bits.

Where:

- <10> Self-Test Fail (STF). The bit is set by the hardware immediately after power up and node reset. When set, this bit indicates that the node has not yet passed its self-test. Operational firmware will clear this bit after self test has completed and passed and RTOS has booted. If self test does not pass, the bit will not be cleared.
- <11> Extended Self-Test Fail (ETF). There is no extended self test for the XFA. Operational firmware must also clear this bit after self test has completed.
- <28> XMI BAD (XBAD). This bit is cleared by operational firmware on power up.

- <29> Node Halt (NHALT). Writing this bit to a one forces the XFA to go into a 'quiet' wait state, looping until the port driver sets Node Reset.

  See Port Shutdown Chapter 5 for more details.
- <30> Node Reset (NRST). Writing a one to this location initiates a complete power-up reset.

  See Port Shutdown Chapter 5 for more details.

2.1.3 XMI Failing Address Register (XFADR)

This register is used to log address and length information associated with a failing XMI transaction. This register is fully described in the XMI specification. (Adapter written, host read)

```
 31 30  29                                                 0
+-----+-----------------------------------------------------+
|     |                  Failing Address                    |
+--^--+-----------------------------------------------------+
   |
   |
   +--- Failing Length (FLN)
```

2.1.4 XMI Failing Address Extension Register (XFAER)

This register is used to log extended address information associated with a failing XMI transaction. This register is fully described in the XMI specification. (Adapter written, host read)

```
 31   28 27 26 25               16 15                         0
+-------+-----+-----------------+------------------------------+
|  CMD  | ZERO| Address Extension |             Mask           |
+-------+-----+-----------------+------------------------------+
```

2.2 XFA Specific Registers

There are a total of fifteen port registers. Eight are needed during normal operation, and seven during power-up, initialization or error recovery.

The registers used during normal operation are:

- Port Ring Control Registers: (XMIT_FL, RCV_FL, CMD_FL, UNSOL_FL)
- Host Hibernation Registers: (XMIT_HIB_LO, XMIT_HIB_HI, RCV_HIB_LO, RCV_HIB_HI)

The registers used during power-up, initialization and error recovery are:

- Port Control Shutdown Register (XPCS)
- Port Control Initialize Register (XPCI)
- Port Status Register (XPST)
- Port Data Register 1 (XPD1)
- Port Data Register 2 (XPD2)

- Power-Up Diagnostic Register (XPUD)
- EEPROM Update Register (EEUP)

2.2.1 Port Ring Control Registers (XMIT_FL, RCV_FL, CMD_FL, UNSOL_FL)

A Port Ring Control Register exists for each host ring defined in this specification. Register writes are performed by the port driver to indicate to the port that one or more entries have changed from host to adapter ownership. After the host puts transmit buffers on the transmit queue for processing it writes the XMIT_FL control register to notify the adapter of work to do. The same is true for all other rings using the appropriate control register. (Host written).

For writes to the Ring Flag Register, data bits <31:1> are not used, bit zero must be set.

The format of the four control registers (one for each ring) is:

```
 3
 1                                                          0
+---------------------------------------------------------+-+
|                                                         |E|
|              Reserved                                   |N|
|                                                         |T|
+---------------------------------------------------------+-+
```

Where:

- Ent. (Entry) Written when the host changes ownership of a ring entry to the adaptor. The port driver need not write this bit for each ownership change, but must do so when finished queueing new entries.

Note: If the host puts a packet on the transmit queue and that packet uses multiple transmit entries then the driver must ensure they change the ownership bit of the first entry after all other entries of that packet have been changed.

2.2.2 Host Hibernation Registers: (XMIT_HIB_LO, XMIT_HIB_HI, RCV_HIB_LO, RCV_HIB_HI)

The function of these registers is to help the XFA prevent unnecessary interrupts to the host by having the XFA monitor host progress in servicing entries on the transmit and receive rings. Further description of the interrupt algorithm that employs these registers is described in Chapter 7 on Port Interrupts (Host written).

It should be noted that for both the receive and transmit hibernation registers it is the write to the low HIB register that issues the status interrupt to the host.

The formats of the two transmit host hibernation registers are:

```
 3
 1                                                          0
+-----------------------------------------------------------+
|                                                           |
|            Host Hibernation Address (31:0)                |
|                                                           |
+-----------------------------------------------------------+
```

```
 3
 1                                           8   7             0
+-----------------------------------------------+----------------+
|                                               |                |
|                                               | Host Hibernation|
|                  Reserved                     |  Address (39:32)|
|                                               |                |
+-----------------------------------------------+----------------+
```

Where:

- Host Hibernation Address. The address of the last ring entry serviced by the port driver prior to hibernation.

Note: For packets which require multiple transmit entries the driver must write the address of the first entry in the hibernation register upon hibernating. For receive packets that are chained the driver will write the address of the last entry processed. In this case it is the entry which has the EOP bit in the Buffer Descriptor set.

The format of the Receive Ring Hibernation Registers (RCV_HIB_LO, RCV_HIB_HI) is identical to the register format for the Transmit Ring Hibernation Registers.

2.2.3 Port Control Registers (XPCS, XPCI)

These registers are used to control initialization and shutdown of the adapter.

These registers are write-only to the port driver. The adapter does not read them, but obtains an indication that they have been written, thus the data written to them is of no consequence.

As the particular register is written, the adapter is commanded as follows:

2.2.3.1 Port Shutdown Command (XPCS)

Shutdown the port, including disabling the FDDI Chip Set. The adapter changes the port state in the Port Status Register from *Running*, *Initialized*, or *Maintenance* to *Uninitialized*. If the SHUTDOWN command is issued while the adapter is in a state other than those specified, adapter firmware will ignore the request. The port driver detects completion of the SHUTDOWN command by this adapter state transition. When shutdown is complete, the port driver should also check the Port Status Register for any error not yet recorded.

The actions done by the adapter on receipt of this command are described in Chapter 5 (Port Shutdown).

Adapter firmware, including RTOS, will remain in control during this transition, i.e. there is not a node processor reset.

The completion of this operation can be detected by the port driver by polling the XPST register for the UNINITIALIZE state. An appropriate timeout value to be used by a port driver is one second.

The format of the XPCS register is:

```
 3
 1                                                            0
+-----------------------------------------------------------+-+
|                                                           |S|
|                        Reserved                           |H|
|                                                           |U|
|                                                           |T|
+-----------------------------------------------------------+-+
```

Where:

- Shut (Host settable). Issues SHUTDOWN command to adapter when set by host.

2.2.3.2 Port Initialize Command (XPCI)

Initialize the adapter. Prior to issuing this command, the physical address of the Port Data Block is supplied in the Port Data Registers. When initialization is complete, the adapter changes the port state in the Port Status Register from *Uninitialized* to *Initialized.* If initialization fails, a port state qualifier is left in the Port Status Register (these qualifiers are listed in Appendix B, and the state remains *Uninitialized.* If the INIT command is issued while the adapter is not in the *Uninitialized* state, adapter firmware will ignore the request.

The format of the XPCI register is:

```
 3
 1                                                            0
+-----------------------------------------------------------+-+
|                                                           |I|
|                          Reserved                         |N|
|                                                           |I|
|                                                           |T|
+-----------------------------------------------------------+-+
```

Where:

- Init (Host settable). Issues Init command to adapter when set by host.

If the INIT command fails, no interrupt is generated and the port driver must detect the failure to initialize by timeout.

The XFA will typically complete this operation in less than one second.

The port driver should not issue a second state command without waiting for the first one to complete. For example, the port driver should wait for a SHUTDOWN command to complete before writing the Port Data Registers with the address of a Port Data Block and issuing an INITIALIZE command. The proper adapter initialization and shutdown sequences are discussed in Chapter 4 (Port Initialization) and Chapter 5 (Port Shutdown).

2.2.4 Port Status Register (XPST)

This register contains current port status. It is used to flag FDDI and adapter state transitions. Also, it can provide indications of the reason for state transitions in the state qualifier field. The three hardware assist bits are used to indicate reset conditions to the port driver. Finally, the EED bit is used to indicate the completion of a load of image code from a host buffer to the adapter EEPROM. (Adapter written, host read)

The port driver can only read this register. The adapter writes this register whenever the port state changes. A write to this register generates a host interrupt for all state changes except RESET and UNINITIALIZE. See chapter 3 for the Port State descriptions. The adapter also writes this register on failed initialization when the states don't change but an error is reported. The register is cleared on power-fail or node reset.

```
 31                          16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
+-----------------------------+-----+---------+-+-+-+-+-----+-+-----+
|                             |R    | LED     |E|S|W|A|R    |F|Port |
|        State Qualifier      |S    | State   |E|P|T|M|S    |R|State|
|                             |V    |         |D|E|O|P|V    |A|     |
|                             |D    |         | | | |E|D    | |     |
+-----------------------------+-----+---------+-+-+-+-+-----+-+-----+
```

Where:

- State Qualifier. This qualifies the state field giving the reason why the adapter is in the particular state it is in.

  Additional data can be given in the Port Data Registers. See Appendix B for a table of state qualifiers and description.

  The state qualifier field is only writable by adapter firmware.
- LED States. This field contains the state of the module LEDs. It's purpose is to allow remote access to the module LED states. For a description see Appendix B.
- EED. Indicates that an adapter EEprom image update has completed. (Adapter written)
- Hardware Assist Bits, set by XFA hardware for fatal errors where the firmware is not reliable. Assertion of any of these bits will result in the immediate reset of the FDDI Chip Set. Whenever any of the following 3 bits are set, the device driver cannot assume the validity of any other bits in this register.
  - SPE. Indicates that a firmware local store parity error has occurred.
  - WTO. Watchdog Timer Timeout has occurred, firmware sanity timer expiration indicates adapter code has reached an unknown state.
  - AMPE. Adapter Processor Subsystem broken, some part of the sub-system logic supporting the adapter processor is unreliable.
- FRA. FDDI RING AVAILABILITY. This bit informs the driver of the accessibility of the FDDI ring. If this bit is 0 then the ring is unavailable and the driver will not accept requests from users to transmit or receive packets. If this bit is 1 then the ring is available and the driver will accept requests to transmit and receive packets. A host interrupt is generated for each state transition of this bit.
- Port state. This is the field the port driver can inspect on interrupt or at some keep alive interval to verify that the port is still operational. Writing this field will generate a host interrupt for all states except for RESETTING and UNINITIALIZE.
  - 0 - *Resetting*, the adapter has just been reset or powered up and is doing self test and power up initialization. All frame processing is disabled in this state.
  - 1 - *Uninitialized*, the adapter has completed reset, power up initialization, or the *Shutdown* command. All frame processing is disabled in this state.
  - 2 - *Initialized*, the adapter is running, processing host command ring entries and unsolicited messages only.

- 3 - *Running*, the adapter is in an operational mode, supporting both the FDDI frame processing, host-command processing and unsolicited ring processing.
- 4 - *Maintenance*, the adapter is in a testing mode, the previous PARAM command has specified a loopback mode.
- 5 - *Halted*, the adapter is in a non-operational mode, caused by the port driver setting the NHALT bit in the XBE Register or as the result of an internal adapter fatal error condition. Exiting this state requires the device driver to issue a node reset.

2.2.5 Port Data Registers (XPD1, XPD2)

These registers are written by the port driver with the physical address of the Port Data Block prior to the INIT state command. They are written by the adapter with the MLA after node reset or node halt. (Adapter or host writable, adapter or host readable) They are also used for one method of EEProm Update Refer to the Diagnostic Operation chapter section on EEPROM updates for a further description.

After power up, or node reset, the adapter writes the Port Data Registers with the MLA in the following form:

```
 31            24 23             16 15                MSB      0
+---------------+-----------------+----------------+----------------+
|  Adr Byte 3   |   Adr Byte 2    |   Adr Byte 1   |   Adr Byte 0   |  XPD1:
+---------------+-----------------+----------------+----------------+

 31                              16 15   LSB                       0
+---------------------------------+----------------+----------------+
|              Zero               |   Adr Byte 5   |   Adr Byte 4   |  XPD2:
+---------------------------------+----------------+----------------+
```

Before issuing an *INIT* state command, the port driver writes the physical address of the Port Data Block to the Port Data Registers in the following form:

```
 31                                 ...                            0
+-------------------------------------------------------------------+
|            Port Data Block Address (bits <31:0>)                  |  XPD1:
+-------------------------------------------------------------------+

 31                        ...                      8 7    ...     0
+--------------------------------------------------+----------------+
|                          MBZ                     | PDB Adr <39:32> |  XPD2:
+--------------------------------------------------+----------------+
```

Note: The starting address of the PDB must always be non-zero, page-aligned.

2.2.6 Power Up Diagnostic Register (XPUD)

This register contains intermediate diagnostic self test results after power up and node reset.

This register is reserved for diagnostic use and can not be written by host software. For more details refer to the diagnostics chapter.

This register is not written by operational firmware. The meaning and use is defined in the XFA Diagnostic Functional/Design Specification.

2.2.7 EEPROM Update Register (EEUP)

This register is written by the host to indicate to the adapter that an EEPROM Update is in progress. Refer to the Diagnostic Operations chapter on EEPROM updates for further description.

For writes to the EEPROM Update Register, data bits <31:1> are not used, bit zero must be set.

The format of the EEUP register is:

```
 3
 1                                                         0
+---------------------------------------------------------+-+
|                                                         |E|
|                      Reserved                           |E|
|                                                         |U|
|                                                         |P|
+---------------------------------------------------------+-+
```

Where:

- EEUP, written with bit zero asserted to commence an EEPROM Update.

2.3 Port Data Block (PDB)

The Port Data Block provides a communications area in host memory between the host and the adapter consisting of initialization data, driver maintained data link counters, port error log data, and port reserved memory. It consists of a single 512-byte block of memory aligned to a 512-byte boundary.

The driver fills in initialization data prior to issuing an *INIT* command. The adapter processes the contents during initialization. If the port driver wishes to change any contents of the port data block, it must shut down and reinitialize the adapter, either supplying a new Port Data Block or the same Port Data Block containing new data.

The driver maintains a set of data link counters on behalf of the adapter firmware, the PDB contains all but one of these counters. In addition, the PDB contains the address of the User Buffer Unavailable (UBUA) Counter, so that the adapter can report this counter when requested.

The adapter writes failure data to the port error log area when a fatal error has been detected. The port driver then uses this data for entries into the host error log.

The remainder of the Port Data Block is reserved for port use. The port will use it for host memory access testing.

The driver may write the counter fields anytime. The adapter can write the error log anytime after receiving the PDB address. The ring and UBUA address and size fields can only be changed by the host prior to initialization of the adapter. The counter fields may be read by the adapter anytime. The host may read the error log anytime.

Figure 2–1: Port Data Block

Figure 2–1 Cont'd on next page

Figure 2–1 (Cont.): Port Data Block

```
 31                              16 15          7     ...     0
+---------------------------------------------------------------+
|                    Command Ring Address (31:00)               | b
+---------------------------------+---------------+           --+
|      Command Ring Size          |     RSVD      |   (39:32)   | b + 4
+---------------------------------+---------------+-------------+
|                    Transmit Ring Address (31:00)              | b + 8
+---------------------------------+---------------+           --+
|      Transmit Ring Size         |     RSVD      |   (39:32)   | b + C
+---------------------------------+---------------+-------------+
|                    Receive Ring Address (31:00)               | b + 10
+---------------------------------+---------------+           --+
|      Receive Ring Size          | Rcv Entry Size |  (39:32)   | b + 14
+---------------------------------+---------------+-------------+
|      Max Receive Frame          |            RSVD             | b + 18
+---------------------------------+-----------------------------+
|                 Unsolicited Ring Address (31:00)              | b + 1C
+---------------------------------+---------------+           --+
|    Unsolicited Ring Size        |     RSVD      |   (39:32)   | b + 20
+---------------------------------+-----------------------------+
|                     Host Status Interrupt                     | b + 24
|       XMI node ID mask          |    Vector <15:4>  | IPL  <3:0>|
+---------------------------------+-------------------+-------+
|  PM RING BUFFER ALLOCATION      |          MBZ <15:1>   |PSBUA| b + 28
+-----------------------------------------------------+-----+
|               Driver  UBUA   Counters Address (31:00)         | b + 2C
+-------------------------------------------------+           --+
|                    MBZ                          |   (39:32)   | b + 30
+-------------------------------------------------+-------------+
|                    SBUA Counter (31:00)                       | b + 34
+---------------------------------------------------------------+
|                    SBUA Counter (63:32)                       | b + 38
+---------------------------------------------------------------+
|                    Octets Received Counter (31:00)            | b + 3C
+---------------------------------------------------------------+
|                    Octets Received Counter (63:32)            | b + 40
+---------------------------------------------------------------+
|                    Octets Sent Counter (31:00)                | b + 44
+---------------------------------------------------------------+
|                    Octets Sent Counter (63:32)                | b + 48
+---------------------------------------------------------------+
|                    Frames Received       (31:00)              | b + 4C
+---------------------------------------------------------------+
|                    Frames Received       (63:32)              | b + 50
+---------------------------------------------------------------+
|                    Frames Sent           (31:00)              | b + 54
+---------------------------------------------------------------+
|                    Frames Sent           (63:32)              | b + 58
+---------------------------------------------------------------+
|                    Multicast Octets Received Counter (31:00)  | b + 5C
+---------------------------------------------------------------+
|                    Multicast Octets Received Counter (63:32)  | b + 60
+---------------------------------------------------------------+
|                    Multicast Octets Sent Counter (31:00)      | b + 64
+---------------------------------------------------------------+
|                    Multicast Octets Sent Counter (63:32)      | b + 68
+---------------------------------------------------------------+
|                    Multicast Frames Received   (31:00)        | b + 6C
+---------------------------------------------------------------+
|                    Multicast Frames Received   (63:32)        | b + 70
+---------------------------------------------------------------+
```

Figure 2–1 Cont'd on next page

Figure 2–1 (Cont.): Port Data Block

```
|                    Multicast Frames Sent  (31:00)                |  b + 74
+------------------------------------------------------------------|
|                    Multicast Frames Sent  (63:32)                |  b + 78
+------------------------------------------------------------------+
|                    Block Check Failure (31:00)                   |  b + 7C
+------------------------------------------------------------------|
|                    Block Check Failure (63:32)                   |  b + 80
+------------------------------------------------------------------+
|                    Frame Status Error (31:00)                    |  b + 84
+------------------------------------------------------------------|
|                    Frame Status Error (63:32)                    |  b + 88
+------------------------------------------------------------------+
|                    Frame Alignment Error (31:00)                 |  b + 8C
+------------------------------------------------------------------|
|                    Frame Alignment Error (63:32)                 |  b + 90
+------------------------------------------------------------------+
|                    Frame Too Long (31:00)                        |  b + 94
+------------------------------------------------------------------|
|                    Frame Too Long (63:32)                        |  b + 98
+------------------------------------------------------------------+
|              Unrecognized Individual Frame Dest (31:00)          |  b + 9C
+------------------------------------------------------------------+
|              Unrecognized Individual Frame Dest (63:32)          |  b + A0
+------------------------------------------------------------------+
|              Unrecognized Multicast Frame Dest (31:00)           |  b + A4
+------------------------------------------------------------------+
|              Unrecognized Multicast Frame Dest (63:32)           |  b + A8
+------------------------------------------------------------------+
|                                                                  |  b + AC
|              Port Error Log Area (128 bytes)                     |
|                                                                  |
+------------------------------------------------------------------+
|                                                                  |  b + 12C
|              RESERVED to Port (212 bytes)                        |
|                                                                  |
+------------------------------------------------------------------+
```

Where:

- Command Ring Address. The physical address of the start of the Command Ring. The ring must be physically contiguous and aligned to a 512-byte boundary.
- Command Ring Size. The total size of the Command Ring in Bytes.
- Transmit Ring Address. The physical address of the start of the Transmit Ring. The Transmit Ring must be physically contiguous and aligned to a 512-byte boundary.
- Transmit Ring Size. The total size of the Transmit Ring in Bytes.
- Receive Ring Address. The physical address of the the start of Receive Ring. The Receive Ring must be physically contiguous and aligned to a 512-byte boundary.
- Receive Ring Size. The total size of the Receive Ring in Bytes.
- Receive Ring Entry Size. The total size of a Receive Ring Entry in Bytes, which must be a integral number of octawords (i.e. a multiple of sixteen).
- Max Receive Frame. The maximum size frame that the port driver will accept, frames larger than this should be discarded by the adaptor.

- Unsolicited Ring Address. The physical address of the start of the Unsolicited Ring. The Unsolicited Ring must be physically contiguous and aligned to a 512-byte boundary.
- Unsolicited Ring Size. The total size of the Unsolicited Ring in Bytes. The maximum size UNSOLICITED ring is 4K bytes.
- Host Status Interrupt. This gives the interrupt information needed by the adapter to generate a host status interrupt. If this longword is zero, no status interrupts will be generated by the adapter. If non-zero, status interrupts will be generated as described in the Miscellaneous chapter.
  - <3:0> IPL. This portion of the interrupt vector represents a four bit mask encoding of the IPL to be used by the adapter for status interrupts. Only one of the four bits should be set for a non-zero vector field, otherwise an error is declared.
  - <15:4> Vector. This corresponds to bits <15:4> of the interrupt vector delivered to the host during the IDENT response.
  - <31:16> Node ID Mask. This is the interrupt destination mask defined on the XMI during the INTR Command, representing XMI nodes 15..0. Multiple bits may be set.

- PSBUA. Potential System Buffer Unavailable bit. This bit is used to notify the device driver that the adapter has failed to fetch a valid receive ring entry buffer. If a fetch fails then PSBUA is set to 1. It is cleared by the device driver who periodically must check this bit and clear it if set. Once set, firmware will not notify the adapter again until a completed fetch of a buffer occurs. Finally, firmware will only set this bit at most once a second. If hardware fails and then subsequently successfully completes a fetch of a buffer multiple times within 1 second, firmware will only set the bit one time. The device driver uses this bit to determine if more receive buffers need to be allocated.
- PM RING BUFFER ALLOCATION. The adapter has approximately 1600 buffers for storing received and transmitted packets internally. This field allows a host to specify the allocation of these buffers between the these two paths. The legal values of between 5 and 95 signifies that the adapter will allocate 5 to 95 percent of available buffers to the receive path. The remaining buffers are allocated to the transmit path. If this field is 0 then allocation is divided equally between the receive and transmit paths.
- UBUA Counter Address. This is the physical address of the quadword UBUA counter maintained by the driver on behalf of the adapter. It represents the number of instances where a buffer request by a user is not filled. This counter is maintained by the port driver but reported by the adapter, thus the need to make this counter visible to the adapter.
- SBUA Counter. A counter maintained by the driver on behalf of the adapter representing the number of times a frame was dropped due to a lack of buffer resources.

- Octets Received Counter. A counter maintained by the driver on behalf of the adapter representing the number of octets received by the driver from the FDDI network, i.e. off the receive ring, since system boot.
- Octets Sent Counter. A counter maintained by the driver on behalf of the adapter representing the number of octets queued by the driver for transmission to the FDDI network since initialization.

- Frames Received Counter. A counter maintained by the driver on behalf of the adapter representing the number of frames received by the driver from the FDDI network, i.e. off the receive ring, since system boot.
- Frames Sent Counter. A counter maintained by the driver on behalf of the adapter representing the number of frames queued by the driver for transmission to the FDDI network since system boot.
- Multicast Octets Received Counter. A counter maintained by the driver on behalf of the adapter representing the number of octets destined for multicast users received by the driver from the FDDI network, i.e. off the receive ring, since system boot.
- Multicast Octets Sent Counter. A counter maintained by the driver on behalf of the adapter representing the number of octets destined for multicast users sent by the driver to the FDDI network since system boot.
- Multicast Frames Received Counter. A counter maintained by the driver on behalf of the adapter representing the number of multicast frames received by the driver from the FDDI network, i.e. off the receive ring, since system boot.
- Multicast Frames Sent Counter. A counter maintained by the driver on behalf of the adapter representing the number of multicast frames queued by the driver for transmission to the FDDI network since system boot.
- Block Check Failure Counter. A counter maintained by the driver on behalf of the adapter representing the number of frames which failed an FCS check.
- Frame Status Error Counter. A counter maintained by the driver on behalf of the adapter representing the number of frames returned with bad status and the CRC was correct.
- Frame Alignment Error Counter. A counter maintained by the driver on behalf of the adapter representing the number of frames received with an alignment error
- Frame Too Long Counter. A counter maintained by the driver on behalf of the adapter representing the number of frames received with an invalid length.
- Unrecognized Individual Frame Destination. A counter maintained by the driver on behalf of the adapter representing the number of frames received on an individual address with no matching data link port.
- Unrecognized Multicast Frame Destination. A counter maintained by the driver on behalf of the adapter representing the number of frames received on an enabled group address with no matching data link port.
- Port Error Log Area. This 128-byte area is written with error data by the adapter on fatal port error resulting in port changing state to HALTed. In the case of mulitple errors detected per instantiation, the adapter loads only information collected on the first error.

  The Port Error Log Area is only written by the adapter after an internal fatal error has been detected. The port driver should initialize this memory to zeros when first initializing the Port Data Block.
- Port Reserved Area. This area is used for accessible host memory for debug, testing purposes and possible future implementations.

2.4 Host Ring Structures

There are four host memory resident rings used for Port Driver-Adapter information exchange. The COMMAND ring is used for host to adapter commands and responses. The TRANSMIT and RECEIVE rings are used exclusively for transmitting frames to and receiving frames from the FDDI network. The UNSOLICITED ring is used for reporting unsolicited internal adapter events to the host. The maximum length of each ring is 4096 bytes. The starting address of each ring must be page aligned.

Each ring is of constant length, which is set at initialization time, and is stored in the Port Data Block.

Within each ring, entries are processed in sequential order, starting from the first ring entry at the given starting address of the ring, returning to the starting address of the ring upon reaching the end of the ring. Also, a single ring entry cannot wrap around from the end to the start of a ring. Both the adapter and the host must process the ring entries in sequential order. However, the adapter may return completed entries to the host out of order with the understanding that the host will not see these until earlier entries have been processed.

When the host owns the ring entry, the adapter can do nothing with the entry but read the entry and check the ownership bit. Similarly, when the adapter owns a ring entry, the host can do nothing with the entry but read the entry and check the ownership bit.

2.4.1 Command Ring

Commands are used to pass information from the host to the adapter. This includes setting parameters, starting and stopping users, and defining and monitoring status of the adapter.

The command ring is a physically contiguous data structure located in host memory aligned to a 512-byte boundary. Each entry is a single quadword, containing the physical address of the allocated buffer.

Command ring entries are processed only when the adapter is in one of the following states: *Initialized*, *Running*, and *Maintenance*.

Each ring entry gives status and buffer segment address information and provides a place to put status information when processing of the ring entry by the adapter is complete. The buffer segment must be physically contiguous.

The buffer described by the ring entry contains a port command. A port command is defined to be any command which is NOT a direct command to transmit data on the FDDI ring.

The buffer pointed to by a command entry consists of a single buffer segment. The buffer segment provided must be of sufficient size to contain both the command to the port and the resultant response.

An individual entry looks like:

Where:

- <31> Status/Own. Ownership bit for this ring entry. If set, the entry is owned by the host. If clear, the entry is owned by the adapter. The ownership bit is cleared by the host when it has set up the ring entry and the host wishes the port to execute the command.

Figure 2–2: Command Ring Entry

```
 31  30  29.....27 26.....22 21.....................9 8...........0
+---+---+----------+---------+-----------------------+------------+
|Own|ERS|    ERC   | Uindex |   Segment Length      | PA_LO (8:0)|
+---+---+----------+---------+-----------------------+------------+
| X |           Buffer Segment Physical Address High  (39:9)      |
+-----------------------------------------------------------------+
```

A ring entry not owned by the host means that no field other than the ownership bit may be considered valid by the host. A ring entry not owned by the adapter means that no field other than the ownership bit may be considered valid by the adapter.

- <30> ERS. Error summary bit for the command. If set, an error has occurred. If clear, then no error has occurred.

  This bit may contain stale data when the host grants ownership of the ring entry to the adapter. The adapter will ignore the stale data and set the correct state of the bit before relinquishing ownership of the entry.
- <29:27> ERC. Error code for this ring entry. If ERS=1, this field will contain a valid error code. See Appendix C for a list of error codes.
- <26:22> UIndex. User Definition Block index. This is the user number assigned to the user by either a previous USTART or the current command is a USTART, containing a new UIndex.
- <21:9> Segment Length. This is the length of the command buffer pointed to by the Buffer Segment Address. The minimum and maximum size of each command is limited by the values given in Appendix D.
- <8:0> PA_LO. Lower 9 bits of the physical address of the Buffer Segment Address
- <31> NIO
- <30:0>Buffer Segment Address. Higher 31 bits of the physical address of the Buffer Segment. Any unused address bits must be given as zero. The format is depicted in the ring entry diagrams.

The User index, Buffer Address, and Segment Length are writable by the driver, readable by the adapter. The ERS and ERC error indications are written by the adapter and read by the host. Command buffers are writable by either the driver or adapter, depending on the specific command.

2.4.1.1 Command Buffer Format

The following sections describe different types of command buffers. A command buffer contains port parameter data used to control the operation of the adapter. A command buffer may also contain maintenance data specific to diagnostic operation. A format requirement for command buffers will be that they start on an octaword boundary. A list of the command opcode values is contained in Appendix A.

2.4.1.1.1 NOP Command

This port command does nothing. The command always succeeds.

The format of the data block follows:

Figure 2–3: NOP Command Format

```
 31                            ...                           0
+-------------------------------------------------------------+
|                         Opcode = NOP                         |
+-------------------------------------------------------------+
```

2.4.1.1.2 SYSID Command

This port command provides host-specific System ID information which will be transmitted by the adapter on a regular basis or upon receipt of a Request ID frame. Some of the SYSID information is provided by the adapter. The remaining fields may be given in this command.

The XFA firmware only accepts and sends SYSIDs on the MLA. SYSID messages may be received on the Remote Console Multicast Address, however these would be handled by the host-resident MOP Portal. The fields provided by the adapter are as follows, with the field values for each (the host may overwrite the Functions and Maintenance parameters): For a complete description of the fields in the SYSID command see the DNA Maintenance Operations Functional Specification.

Table 2–1: SYSID Fields Supplied By Adapter

Field	MOP Field Type	Length in Bytes	Value and/or Meaning
Maintenance Version	1	3	04-00-00, indicating MOP Version 4 (802 frames)
Station ID	13	8	MLA, FDDI Physical Address of the Station. The high 2 order bytes are 0
Functions	2	2	51-00, V4 Maintenance functions currently available which indicates 'boot', 'data link counters', and 'loop'
Hardware Address	7	6	MLA (from the Default Physical Address ROM on the module)
Communications Device	100	1	X^5D, Device Type
Communications Device Related	101	2	YYXX, Adapter revision field. XX, the hardware revision; YY, the firmware revision.
Data Link	400	1	5 (FDDI), type of data link protocol

See the DNA Maintenance Operations Functional Specification for more information concerning each field.

The format of the data block follows:

Figure 2–4: SYSID Command Format

```
 31                          ...                          0
+----------------------------------------------------------+
|                      Opcode = SYSID                      |
+----------------------------------------------------------+
|                                                          |
|                                                          |
|                        SYSID Data                        |
|                                                          |
|                                                          |
+----------------------------------------------------------+
```

2.4.1.1.3 PARAM Command

This command is used to modify various operating parameters and to reinitialize values (especially counters) which need to be saved across adapter resets. The Adapter Manager updates its internal storage of the Adapter variables, time, and counters passed in the Command Buffer, initiates a connection attempt to the ring, reinitializes all counter values which may have been lost across an adapter reset, and finally changes the Adapter State to either Running or Maintenance depending upon the command requirement. This command only succeeds when issued in the Initialized State, otherwise an Invalid Command error is reported.

The Adapter Manager uses the second section of the PARAM command buffer to periodically store a copy of it's internal counters. The period is determined in the command itself. Across adapter resets, these counter values are then read by the Adapter Manager to restore their original values. This provides the network management a better picture of network activity during an individual system boot.

The format of the PARAM Command Buffer is:

Figure 2–5: PARAM Command Format

```
 31           ...            16 15           ...          0
+----------------------------------------------------------+
|                      Opcode = PARAM                      | b
+----------------------------------------------------------+
|                          Version                         | b + 4
+----------------------------------------------------------+
|                            MLA                           | b + 8
+-----------------------------+                          --+
|            RSVD             |                            |
+-----------------------------+----------------------------+
|                           T_MAX                          | b + 10
+----------------------------------------------------------+
|                           T_REQ                          | b + 14
+----------------------------------------------------------+
|                            TVX                           | b + 18
+----------------------------------------------------------+
|                       LEM Threshold                      | b + 1C
+----------------------------------------------------------+
|                                                          | b + 20
+--                            BVC                       --+
|                                                          |
+----------------------------------------------------------+
|                          Max_User                        | b + 28
```

Figure 2–5 Cont'd on next page

Figure 2–5 (Cont.): PARAM Command Format

```
+------------------------------------------------------------+
|                        Max_Adr                             | b + 2C
+-----------------------------------------+------+---+---+
| RSVD <31:4>                             |DIS RP|BOO|LM | b + 30
+---------------------------+-------------+------+---+---+
| RTOKEN Timeout            | Update Period (in seconds) | b + 34
+---------------------------+--------------------------+
|                  Time Since Last Init                | b + 38
|              (binary relative format)                |
+------------------------------------------------------+
|          Line Counters:  (all quadword)              | b + 48
|                                                      |
|                                                      |
|       Octets Received                                |
|       Octets Sent                                    |
|       Frames Received                                |
|       Frames Sent                                    |
|       Multicast Octets Received                      |
|       Multicast Octets Sent                          |
|       Multicast Frames Received                      |
|       Multicast Frames Sent                          |
|       Transmit Underrun                              |
|       Transmit Failures                              |
|       Receive Failures - Block check failure         |
|       Receive Failures - Frame status error          |
|       Receive Failures - Frame alignment error       |
|       Receive Failures - Frame too long              |
|       Unrecognized Indiv address Frame Destination   |
|       Unrecognized Multi address Frame Destination   |
|       Receive Overrun                                |
|       Link Buffer Unavailable                        |
|       User Buffer Unavailable                        |
|       Received Frame Count                           |
|       Error Count                                    |
|       Lost Count                                     |
|       Ring Initialization - Initiated                |
|       Ring Initialization - Other Station            |
|       Duplicate Address Test failed                  |
|       Duplicate Token Detected                       |
|       Ring Purge Errors                              |
|       Bridge_strip Errors                            |
|       PC Traces Initiated                            |
|       Link Self Test Failures                        |
|       LEM Rejects                                    |
|       LEM Link Errors                                |
|       LCT Rejects                                    |
|       TNE EXP Rejects                                |
|       Ebuff errors                                   |
|       Connections Completed                          |
|       PC Traces Received                             |
|       Reserved Counter                               |
+------------------------------------------------------+
```

The Host fills in all fields except MLA, Max_Adr, Max_User, and Version. The data written by the Host in these fields is ignored by the adapter and replaced in the command buffer response by the actual values known by the port.

Where:

- Version. Identifies the version of the Datalink Architecture supported by this adapter.

- MLA (formerly DPA). My Long Address. FDDI Physical Address. The default physical address assigned to the adapter (stored in the MAC Address ROM).
- T_Max. Maximum token rotation time. (Default = MIN = 173.01504 ms). Granularity of 80ns. If zero is provided in this field, the default value is used by CNS.
- T_Req. Requested token rotation time. (T_Min <= T_Req <= T_Max, Default = 8.00 ms, T_MIN = 4ms) Granularity of 80ns. If zero is provided in this field, the default value is used by CNS.
- TVX. Valid Transmission Time. (2.35 Msec <= TVX <= 5.2224 Msec, Default = 2.62144ms) Granularity of 80ns. If zero is provided in this field, the default value is used by CNS.
- LEM Threshold. (8 >= LEM_Threshold >= 5, Default = 8) The value of the field represents a negative exponent, i.e. 5 represents a link error threshold of 10**-5. If zero is provided in this field, the default value is used by CNS.
- BVC. 8 byte boot verification code to be compared with the verification code contained in boot messages to determine their validity.
- Max_User. Maximum number of Users supported by the adapter. Written by the adapter. Equals 32 for XFA. However user index 31 is reserved for the Adapter Manager. User index 30 is reserved for the SMT application/SMT management layer. User index 29 is reserved for the UNKNOWN user.
- Max_Adr. Maximum number of filter addresses of all types supported by the adapter. Written by the adapter. Equals 64 for XFA. The 64th entry is reserved for the AMC user.
- <1:0> Loopback Mode. Written by the Host. The value of this two bit field determines whether the adpater enters the Maintenance State or not, and which Loopback mode is enabled. The Maintenance State indicates that the adpater is one of several Loopback states, which are listed below.
  - 0 - No loopback, Adapter enters the Running State.
  - 1 - External Loopback. MAC chip is put in copy self mode. Adapter enters the Maintenance State.
  - 2 - Loopback between the Receive and Transmit Clock and Data Conversion chips. MAC chip is put in copy self mode. Adapter enters the Maintenance State.
  - 3 - Reserved.
- <2> BOO. Boot Enable Flag. This is the Boot Enable Flag. If set then firmware processes received boot messages. If clear and there is a 60-02 user then firmware sends the frame to the host, destined for that user. Written by the Host.
- <3> DIS RP. Disable Ring Purger. If set the firmware will disable the ring purger. If clear, it will allow this node to particpate in the ring purger selection.
- <15:00> UPDATE PERIOD. Specifies the number of seconds in which firmware will update the counters block with a copy of its internal counters. A value of 0 disables any periodic update.
- <31:16> RTOKEN Timeout. Restricted Token Timeout. This timeout can be set by the driver. The MAX value is 10 seconds. It should be passed to the adapter in 160usec units. i.e. 10 secs = 62500.

- Time Since Last Init. This value is the time in binary relative format since the last system boot was done. The Port Driver must initialize this value with the correct time since system boot before issuing the PARAM command. The timer field is actually represented in 100 Nsec units, however, the accuracy of the field is one second.
- Counter Block. Holds all adapter internally stored counters. This is used to enable reiniting the counter values across adapter resets. Firmware periodically stores its counter values to enable reiniting them across resets.

2.4.1.1.4 STATUS Command

This port command checks FDDI parameters, such as dual-address test results and other non-network characteristics. This information is returned in the response as shown in the buffer diagram.

The STATUS command can be issued only when the adapter is in the *Running* or *Maintenance* states.

The format of the status data block follows:

Figure 2–6: STATUS Command Format

```
 31           ...          16 15          ...            0
+--------------------------------------------------------+
|                   Opcode = STATUS                      | b
+--------------------------------------------------------+
|                 Time Since Last Init                   | b + 4
|             (binary relative format)                   |
+--------------------------------------------------------+
|                       Version                          | b + 14
+--------------------------------------------------------+
|                         MLA                            | b + 18
+-----------------------------+                        --+
|           RSVD              |                          |
+-----------------------------+--------------------------+
|                        T_MAX                           | b + 20
+-----------------------------+--------------------------+
|                        T_NEG                           | b + 24
+-----------------------------+--------------------------+
|                         TVX                            | b + 28
+-----------------------------+--------------------------+
|                   LEM Threshold                        | b + 2C
+-----------------------------+--------------------------+
|                     UN Address                         | b + 30
+-----------------------------+                        --+
|           RSVD              |                          |
+-----------------------------+--------------------------+
|                                                        | b + 38
+--                      BVC                           --+
|                                                        |
+--------------------------------------------------------+
|                       Cur_Dst                          | b + 40
+--------------------------------------------------------+
|                       Cur_User                         | b + 44
+--------------------------------------------------------+
|                       Cur_Mca                          | b + 48
+--------------------------------------------------------+
|                       Max_User                         | b + 4C
+--------------------------------------------------------+
|                       Max_Adr                          | b + 50
+--------------------------------------------------------+
|                      PHY STATE                         | b + 54
+--------------------------------------------------------+
|                      Port Type                         | b + 58
+--------------------------------------------------------+
|                                                        | b + 5C
+--                                                    --+
|                 Module Serial Number                   |
+--                                                    --+
|                                                        |
+-------------------+---+------+-----+---------+--+---+--+---+
| RSVD <31:13>      |FDX|FDX_EN|RP_EN|RSVD<9:8>|RP|DAT|ST|BOO| b + 68
+-------------------+---+------+-----+---------+--+---+--+---+
```

Note: The data written by the port driver to any field is ignored by the adapter and replaced in the command buffer response by the actual values known by the port.

Where:

- Time Since Last Init. This value is the time in binary relative format since the last system boot was done. The timer field is actually represented in 100 Nsec units, however, the accuracy of the field is one second.

- Version. Identifies the version of the data link architecture supported by this adapter.
- MLA. "My Long Address", this is the default physical address assigned to the adapter (stored in the Default Physical Address ROM).
- T_Max. Maximum token rotation time. (Default = MIN = 173.01504 ms). Granularity of 80ns.
- T_Neg. Negotiated token rotation time. (4 Msec <= T_Neg <= T_Max) Granularity of 80ns.
- TVX. Valid Transmission Time. (2.35 Msec <= TVX <= 5.2224 Msec) Granularity of 80ns.
- LEM Threshold. (8 >= LEM_Threshold >= 5) The value of the field represents a negative exponent, i.e. 5 represents a link error threshold of 10**-5.
- UN Address. Upstream Neighbor Address
- BVC. 8 byte boot verification code compared to the verification code contained in boot messages for determining their validity.
- Cur_Dst. Current number of filter addresses defined among the various User Definition Blocks for physical destination address filtering (part of the Receive Address Filter Data Base.
- Cur_User. Current number of User Definition Blocks defined to the adapter.
- Cur_Mca. Current number of filter addresses defined among the various User Definition Blocks for multicast destination address filtering.
- Max_User. Maximum number of Users supported by the adapter.
- Max_Adr. Maximum number of filter addresses of all types supported by the adapter.
- PHY STATE. The state of the PHY link when the Status command was received.
- Station Type. The Station Type of the Adapter. SAS = 0
- Module Serial Number. This field is ACSII encoded, the LSB is at offset 0x5C, and the MSB is at offset 0x67. It's a ten byte field so the encoding is two padded bytes of zero (ASCII), then the site code, week number and board number.
- <0> BOO. Boot Enable Flag. If the BOO flag is set, remote boot message processing is enabled. If the BOO flag is clear, then boot messages are only passed to the host if there is an 60-02 user in the host.
- <3:1> ST. FDDI State, current state of link attachment when the STATUS response is generated.
  - 0 - Off Init
  - 1 - Off Ready
  - 2 - Off Fault Recovery
  - 3 - On Ring Init
  - 4 - On Ring Run
  - 5 - Broken
- <5:4> DAT. Dual Address Test Results, which will be:
  - 0 - Unknown

  - 1 - Passed
  - 2 - Failed
  - 3 - Reserved
  - <7:6> RP. Ring Purger State, which will be:
  - 0 - Purger Off
  - 1 - Candidate
  - 2 - Non Purger
  - 3 - Purger
- <10>RP EN. Ring Purger Enabled. This bit is set if the ring purger is enabled to run. If clear, the ring purger is off.
- <11> FDX EN. Full Duplex Enabled. If set, the DEMFA is enabled to run in FDX mode whenever there are two nodes connected point to point and the other node has FDX enabled also.
- <12> FDX MODE. Full Duplex Mode. If set, the DEMFA is running in FDX mode.

2.4.1.1.5 RDCNTR Command

This port command reads port counters.

In DECNET Phase V counters are not cleared. Counters are only cleared in XFA when the host reboots. Since quadword counter values are maintained, there is no danger of overflow and hence no need to clear them. The MOP Version 4 counters message contains values representing uncleared counter values.

The counter values returned by this command represent the sum of adapter maintained counters and the host maintained counters from the PDB.

The format of the counters command buffer follows, where each field except for the opcode field is of quadword length. The meaning of each of the counters values is derived from the XI Data Link Architecture Specification.

The Time Since Last Init field is the time in binary relative format since the last system boot was done. It represents the time over which the counter values have incremented. The timer field is actually represented in 100 Nsec units, however, the accuracy of the field is one second.

The entire command buffer is written by the adapter and read by the driver.

Figure 2–7: RDCNTR Command Format

Figure 2–7 Cont'd on next page

Figure 2–7 (Cont.): RDCNTR Command Format

```
 31                          ...                          0
+---------------------------------------------------------+
|                  Opcode =   RDCNTR                       | b
+---------------------------------------------------------+
|                    Time Since Last Init                 | b + 4
|                (binary relative format)                 |
+---------------------------------------------------------+
|        Octets Received (31:0)                           | b + 14
+---------------------------------------------------------+
|        Octets Received (63:32)                          |
+---------------------------------------------------------+
|        Octets Sent     (31:0)                           | b + 1C
+---------------------------------------------------------+
|        Octets Sent (63:32)                              |
+---------------------------------------------------------+
|        Frames Received (31:0)                           | b + 24
+---------------------------------------------------------+
|        Frames Received (63:32)                          |
+---------------------------------------------------------+
|        Frames Sent     (31:0)                           | b + 2C
+---------------------------------------------------------+
|        Frames Sent (63:32)                              |
+---------------------------------------------------------+
|        Multicast Octets Received (31:0)                 | b + 34
+---------------------------------------------------------+
|        Multicast Octets Received (63:32)                |
+---------------------------------------------------------+
|        Multicast Octets Sent  (31:0)                    | b + 3C
+---------------------------------------------------------+
|        Multicast Octets Sent (63:32)                    |
+---------------------------------------------------------+
|        Multicast Frames Received (31:0)                 | b + 44
+---------------------------------------------------------+
|        Multicast Frames Received (63:32)                |
+---------------------------------------------------------+
|        Multicast Frames Sent  (31:0)                    | b + 4C
+---------------------------------------------------------+
|        Multicast Frames Sent (63:32)                    |
+---------------------------------------------------------+
|        Send Failures - Transmit Underrun  (31:0)        | b + 54
+---------------------------------------------------------+
|        Send Failures - Transmit Underrun (63:32)        |
+---------------------------------------------------------+
|        Transmit Failures (31:0)                         | b + 5C
+---------------------------------------------------------+
|        Transmit Failures (63:32)                        |
+---------------------------------------------------------+
|        Receive Failures - Block Check Error (31:0)      | b + 64
+---------------------------------------------------------+
|        Receive Failures - Block Check Error (63:32)     |
+---------------------------------------------------------+
|        Receive Failures - Frame Status Error (31:0)     | b + 6C
+---------------------------------------------------------+
|        Receive Failures - Frame Status Error (63:32)    |
+---------------------------------------------------------+
|        Receive Failures - Frame Alignment Error (31:0)  | b + 74
+---------------------------------------------------------+
|        Receive Failures - Frame Alignment Error (63:32) |
+---------------------------------------------------------+
|        Receive Failures - Frame Too Long (31:0)         | b + 7C
```

Figure 2–7 Cont'd on next page

Figure 2–7 (Cont.): RDCNTR Command Format

Receive Failures - Frame Too Long (63:32)	
Unrecognized Indiv Frame Destination (31:0)	b + 84
Unrecognized Indiv Frame Destination (63:32)	
Unrecognized Multi Frame Destination (31:0)	b + 8C
Unrecognized Multi Frame Destination (63:32)	
Receive Failures - Data Overrun (31:0)	b + 94
Receive Failures - Data Overrun (63:32)	
Receive Failures - Link Buffer Unavailable (31:0)	b + 9C
Receive Failures - Link Buffer Unavailable (63:32)	
Receive Failures - User Buffer Unavailable (31:0)	b + A4
Receive Failures - User Buffer Unavailable (63:32)	
Frame Count (31:0)	b + AC
Frame Count (63:32)	
Error Count (31:0)	b + B4
Error Count (63:32)	
Lost Count (31:0)	b + BC
Lost Count (63:32)	
Ring Initialization - Initiated (31:0)	b + C4
Ring Initialization - Initiated (63:32)	
Ring Initialization - Other Station (31:0)	b + CC
Ring Initialization - Other Station (63:32)	
Duplicate Address Test Failed (31:0)	b + D4
Duplicate Address Test Failed (63:32)	
Duplicate Token Detected (31:0)	b + DC
Duplicate Token Detected (63:32)	
Ring Purge Error (31:0)	b + E4
Ring Purge Error (63:32)	
Bridge Strip Error (31:0)	b + EC
Bridge Strip Error (63:32)	
PC Traces Initiated (31:0)	b + F4

Figure 2–7 Cont'd on next page

Figure 2–7 (Cont.): RDCNTR Command Format

```
|        PC Traces Initiated (63:32)                          |
+-------------------------------------------------------------+
|        Link Self Test Error (31:0)                          | b + FC
+-------------------------------------------------------------+
|        Link Self Test Error (63:32)                         |
+-------------------------------------------------------------+
|        LEM_Rejects (31:0)                                   | b + 104
+-------------------------------------------------------------+
|        LEM_Rejects (63:32)                                  |
+-------------------------------------------------------------+
|        LEM_Link_Errors (31:0)                               | b + 10C
+-------------------------------------------------------------+
|        LEM_Link_Errors (63:32)                              |
+-------------------------------------------------------------+
|        LCT_Rejects (31:0)                                   | b + 114
+-------------------------------------------------------------+
|        LCT_Rejects (63:32)                                  |
+-------------------------------------------------------------+
|        TNE EXP Rejects (31:0)                               | b + 11C
+-------------------------------------------------------------+
|        TNE EXP Rejects (63:32)                              |
+-------------------------------------------------------------+
|        ELM_Parity_Errors (31:0)                             | b + 124
+-------------------------------------------------------------+
|        ELM_Parity_Errors (63:32)                            |
+-------------------------------------------------------------+
|        Connections_Completed (31:0)                         | b + 12C
+-------------------------------------------------------------+
|        Connections_Completed (63:32)                        |
+-------------------------------------------------------------+
|        PC Traces Received (31:0)                            | b + 134
+-------------------------------------------------------------+
|        PC Traces Received (63:32)                           |
+-------------------------------------------------------------+
|        Reserved Counter (31:0)                              | b + 13C
+-------------------------------------------------------------+
|        Reserved Counter (63:32)                             |
+-------------------------------------------------------------+
```

2.4.1.1.6 USTART Command

This port command defines user definition data. Defining each data link user according to the PID's, 802 SAP, 802 SNAP values and associated multicast addresses allows the port driver to categorize received frames for the port and to dictate to the adapter how to do frame filtering.

The User_Index given in the command ring entry gives the unique identifier to be used whenever the port driver or the adapter refers to the user. The User_Index specifies an index ranging from 0 to 30. User index 31 in used internally by the adapter. User index 30 is reserved for the SMT application/SMT management layer. User index 29 is reserved for the UNKNOWN user. Finally, the user specified must not already exist; if it does, the command will fail with 'Invalid command' status in the command entry error field.

The command buffer is read by the adapter, written by the driver.

The format of the USTART follows:

Figure 2–8: USTART Command Format

```
 31                             16  15                            0
+----------------------------------------------------------------+
|                          Opcode = USTART                       | B
+-----------------------------------+--+----+---+---+----+---+-------+
|                 RSVD              |DE|IFCS|CL1|AMC|RSVD|P/S| MODE  | B + 4
+-----------------------------------+--+----+---+---+----+---+-------+
|                       RSVD                     |       FC        | B + 8
+------------------------------------------------+----------------+
|                  User Physical Address (47:00)                  | B + C
+--------------------------------+                              --+
|             RSVD               |                                |
+--------------------------------+--------------------------------+
|             RSVD               |             Adr_Len            | B + 14
+--------------------------------+--------------------------------+
|                          Address 1 (47:00)                     | B + 18
+--------------------------------+                              --+
|             RSVD               |                                |
+--------------------------------+--------------------------------+
                                .....

+--------------------------------+--------------------------------+
|                          Address 16 (47:00)                    | B + 90
+--------------------------------+                              --+
|             RSVD               |                                |
+---------------+----------------+----------------+---------------+
|    Gsap 3     |    Gsap 2      |     Gsap 1     |      ISAP     | B + 98
+---------------+----------------+----------------+---------------+
|    Gsap 7     |    Gsap 6      |     Gsap 5     |     Gsap 4    | B + 9C
+---------------+----------------+----------------+---------------+
|                             PID   (39:00)                       | B + A0
+--------------+-----------------+----------------+             --+
|             RSVD                                |               | B+ A4
+----------------------------------------------------------------+
```

Where:

- Second Longword:
  - <1:0> MODE. Filtering Mode for the Adapter.
    - 0 = Normal mode. Full filtering is done. The level of filtering is dependent on the type of frames requested. For instance SMT frames are filtered on FC and DA only while LLC frames are filtered on FC, DA and DSAP/PIDs.
    - 1 = FC filtering only ("promiscuous mode"). All other fields are ignored.
    - 2 = RSVD
    - 3 = RSVD.
  - <2> P/S. If set, the Ustart PID field is valid. If clear, the SAP field is valid.
  - <3> Reserved.
  - <4> AMC. If *set*, frames addressed to any multicast address which pass the LLC filtering of this user are accepted on behalf of this user. This bit can only be enabled for LLC users.

- <5> CL1. If set, the user being defined is a Class 1 user, indicating that the data link handles the entire LLC protocol. If clear, the data link handles only MLA and LLC SAP address filtering, the user must perform the remainder of the filtering.
- <6> IFCS. Ignore FCS errors. If set, ignore FCS errors for this user.
- <7> DE. Deliver error packets. If set, deliver error packets for this user.
- <31:8> Reserved.

- Third Longword:
  - <7:0> FC. Frame Control. Defines the range of Frame Control field values this User wishes to be delivered. This field consists of bit switches, indicating which range(s) of FC values are to be delivered to this user. All users with the exception of the promiscuous user can only enable one type of frame control. Bits within this field are defined as follows:
    - Bit 0: Void Frames. Not settable, MBZ.
    - Bit 1: Non-Restricted Token. Not settable, MBZ.
    - Bit 2: Restricted Token. Not settable, MBZ.
    - Bit 3: SMT frames. Settable. (FC and DA filtering only for non promiscuous
    - Bit 4: MAC frames. Not settable.
    - Bit 5: LLC frames. Settable.
    - Bit 6: Reserved for implementor frames. Settable. (FC and DA filtering only for non promiscuous users)
    - Bit 7: Reserved for future standardization frames. Settable. (FC and DA filtering only for non promiscuous users)
  - <31:8> Reserved.
- User Physical Address. This is the physical address which associates the user with the station. If non-zero, the adapter accepts frames for this user using the UPA as an 'alias' address. It does not effect the use of the MLA for any other user. If zero or equal to the MLA, the adapter accepts frames for this user on the MLA address.
- Adr_Len. This is the actual number of multicast addresses the host is providing for this user. The maximum number of addresses supported for any single user is 16.
- Address 1-16. Multicast addresses.
- ISAP. 802 frames which are addressed to this SAP value are accepted on behalf of this User Definition Block. This field is valid only if the P/S bit is clear, and it may never equal the SNAP value of "AA". If the P/S bit is clear, the ISAP value must be specified.
- GSAPs 1..7. These are similar to SAP identifiers for FDDI users. 802 frames which are addressed to one of these Group Sap values are accepted on behalf of this User Definition Block. In addition, they are delivered to any other user that has also enabled this GSAP. If a GSAP field is non-zero and the I/G-bit is set, it is considered valid. These fields are valid only if the P/S bit is clear. Any frame delivered to the Port Driver that uses a Group Sap will have the Multiple Destination bit set in the Receive Ring entry.
- SNAP PID. This is the protocol ID for frames addressed to this user. This field is valid only if the P/S bit is set.

2.4.1.1.7 USTOP Command

This port command undefines user definition data. The user to be deleted is identified by the User_Index given in the command ring entry.

The format of the command is as follows:

Figure 2–9: USTOP Command Format

```
 31                              ...                              0
+-----------------------------------------------------------------+
|                          Opcode = USTOP                         |
+-----------------------------------------------------------------+
```

2.4.1.1.8 UCHANGE Command

This port command redefines user definition data for an existing user. It is identical to a USTOP command followed by a USTART command (aside from the opcode). The format of the command is the same as the USTART command. Any fields, except the Uindex may be modified.

2.4.1.1.9 SET SMT Command

This port command allows the driver to set SMT settable parameters. It will be used after the PARAM command has been issued and only while the port is in the RUNNING state. The appropriate values are modified and then a ring reinitialization is performed to instantiate the changes.

Figure 2–10: SET SMT Command Format

```
 31           ...           16 15           ...             0
+-----------------------------------------------------------+
|                       Opcode = SET_SMT                    | b
+-----------------------------------------------------------+
|                     Parameter valid mask                  | b + 4
+-----------------------------------------------------------+
|                          T_MAX                            | b + 8
+-----------------------------------------------------------+
|                          T_REQ                            | b + C
+-----------------------------------------------------------+
|                           TVX                             | b + 10
+-----------------------------------------------------------+
|                       LEM Threshold                       | b + 14
+-----------------------------------------------------------+
|                      RTOKEN Timeout                       | b + 18
+---------------------------------------------------+-------+
|                      RESERVED<31:1>               |FDX_EN | b + 1C
+---------------------------------------------------+-------+
```

- Parameter valid mask. If appropriate bit is set then use the values provided in the corresponding location in the command, otherwise ignore that field in the command.
  1. Bit 0: Disable Ring Purger field is valid
  2. Bit 1: T_REQ field is valid
  3. Bit 2: TVX field is valid
  4. Bit 3: LEM Threshold is valid

5. Bit 4: Restricted Token Timeout field is valid
6. Bit 5: FDX field is valid
7. All other bits ignore

- <0> Disable Ring Purger. If set the firmware will disable the ring purger. If clear, it will allow this node to particpate in the ring purger selection.
- T_Req. Requested token rotation time. (T_Min <= T_Req <= T_Max, Default = 8.00 ms, T_MIN = 4ms) Granularity of 80ns.
- TVX. Valid Transmission Time. (2.35 Msec <= TVX <= 5.2224 Msec, Default=2.62144ms)) Granularity of 80ns.
- LEM Threshold. (>=10**-8 and <=10**-5, Default = 8) Granularity of 80ns. The value of the field will be within the range 5 through 8. Firmware will store the appropriate value in registers.
- RTOKEN Timeout. Restricted Token Timeout. This timeout can be set by the driver. The MAX value is 10 seconds. It should be passed to the adapter in 160usec units. i.e. 10 secs = 62500.
- <0> FDX. Enable Full Duplex Mode. If set the firmware will allow the DEMFA to operate in Full Duplex mode whenever there are two nodes connected point to point.

2.4.1.1.10 Maintenance Commands

Maintenance Commands are handled by the Adapter Manager similarly to other Commands, except that the length in bytes of any adapter response entered into the command buffer is entered into the Command Ring Entry BUFFER LENGTH field before ownership is returned to the Host.

The format of the command is as follows :

Figure 2–11: MAINT Command Format

```
 31                          ...                             0
+-------------------------------------------------------------+
|                       Opcode = MAINT                        |
+------------------------------+------------------------------+
|            Status            |       Maintenance Opcode     |
+------------------------------+------------------------------+
|                                                             |
:             Maintenance Data (adapter specific)             :
|                                                             |
+-------------------------------------------------------------+
```

Where:

- Maintenance Opcode. Maintenance function, see below.
- Status. Maintenance function status, returned on completion of the command, given in Appendix C. This status is relative to the maintenance operation itself, not to the generic execution of a command. If the adapter is able to read the command buffer, the command opcode indicates "MAINT command" and the command buffer is at least 8 bytes long (long enough for a command opcode, maintenance opcode and maintenance

status), the command status is 'no error'. Any subsequent errors, such as buffer length error, appear as maintenance status, not command ring entry status.

Maintenance commands are defined for the following purposes:

- Debugging adapter firmware and hardware
- Reading and writing EEPROM contents
- Reading MOP Boot message data
- Reading module status data

2.4.1.1.10.1 Debug Commands

Used to assist in debugging adapter firmware, the *debug* command is a generic command to check the state of adapter firmware or to provide external stimulus.

Figure 2–12: Debug Command Format

```
 31                               ...                              0
+------------------------------------------------------------------+
|                        Opcode = MAINT                            |
+------------------------------+-----------------------------------+
|            Status            |      MaintOpcode = Debug          |
+------------------------------+-----------------------------------+
|           Reserved           |      Debug Opcode                 |
+------------------------------+-----------------------------------+
|                  Firmware Data or Status                         |
|                                                                  |
+------------------------------------------------------------------+
```

2.4.1.1.10.1.1 Read$Status Debug Command

On receipt of this debug command the Adapter Manager writes module status information maintained for debugging purposes to the Command Buffer. This command is available for diagnostic use and possible future use by a network management entity running on the system.

The command format follows. Only the opcode field is supplied when the command is issued. The remaining fields are filled in by the Adapter Manager when executing the command.

The Time Since Last Init field is the time in binary relative format since the last system boot was done. It does not represent the time over which the hardware discard counter values have incremented, since the adapter may have been reset during this time interval. The timer field is actually represented in 100 Nsec units, however, the accuracy of the field is one second.

```
 31                              ...                                0
+-----------------------------------------------------------------+
|                        Opcode = MAINT                           | b
+--------------------------------+--------------------------------+
|             Status             |     MaintOpcode = Debug        | b + 4
+--------------------------------+--------------------------------+
|           Reserved             |     Debug_Cmd = Read$Status    | b + 8
+--------------------------------+--------------------------------+
|        Device Revision         |          Device Type           | b + C
+--------------------------------+--------------------------------+
|                      Time Since Last Init                       | b + 10
|                  (binary relative format)                       |
+-----------------------------------------------------------------+
|                     ESP Error Register                          | b + 20
+-----------------------------------------------------------------+
|                     XMI Bus Error Register                      | b + 24
+-----------------------------------------------------------------+
|                     XMI Failing Address Register                | b + 28
+-----------------------------------------------------------------+
|                  XMI Failing Address Extension Register         | b + 2C
+-----------------------------------------------------------------+
|                  Power-Up Diagnostic Register                   | b + 30
+-----------------------------------------------------------------+
|                  AM Interrupt Register                          | b + 34
+-----------------------------------------------------------------+
|             68020 Stack, 16 longwords                           | b + 38
+-----------------------------------------------------------------+
|                      Condition Code Register (CCR)              | b + 78
+-----------------------------------------------------------------+
|                      Interrupt Stack Pointer (ISP)              | b + 7C
+-----------------------------------------------------------------+
|                      Master Stack Pointer (MSP)                 | b + 80
+-----------------------------------------------------------------+
|                      Status Register (SR)                       | b + 84
+-----------------------------------------------------------------+
|                      Vector Base Register (VBR)                 | b + 88
+-----------------------------------------------------------------+
|                      Function Code Source (SFC)                 | b + 8C
+-----------------------------------------------------------------+
|                      Function Code Destination (DFC)            | b + 90
+-----------------------------------------------------------------+
|                      Cache Control Register (CACR)              | b + 94
+-----------------------------------------------------------------+
|                      Cache Address Register (CAAR)              | b + 98
+-----------------------------------------------------------------+
|         PMC Hardware Discard Counters (one longword each)       | b + 9C
|                                                                 |
|                 Parser Packet Discard Count                     |
+-----------------------------------------------------------------+
|                   Host Packet Discard Count                     | b + A0
+-----------------------------------------------------------------+
|                     AM Packet Discard Count                     | b + A4
+-----------------------------------------------------------------+
|                    SMT Packet Discard Count                     | b + A8
+-----------------------------------------------------------------+
|                    MOP Packet Discard Count                     | b + AC
+-----------------------------------------------------------------+
|               Oversize Packet Discard Count                     | b + B0
+-----------------------------------------------------------------+
|                  Error Packet Discard Count                     | b + B4
+-----------------------------------------------------------------+
```

2.4.1.1.10.2 Clear Error Log Command

This command is used to clear out the Adapter Error Log of any error data. Only the data is cleared. The counters that keep track of the number of error log writes is incremented and saved.

```
 31                             ...                             0
+---------------------------------------------------------------+
|                        Opcode = MAINT                         |
+-------------------------------+-------------------------------+
|            Status             |  MaintOpcode = Clear Errorlog |
+-------------------------------+-------------------------+-----+
```

2.4.1.1.10.3 Read Error Log Command

This command is used to read the Adapter Error Log. The whole Error Log is read. There are no provisions to read an individual Error Log entry. Also given along with the Error Log is the count specifying the number of times each Error Log entry was written as well as a CRC for each entry. Currently there are 4 error log entries for the Adapter Error Log.

```
 31                             ...                             0
+---------------------------------------------------------------+
|                        Opcode = MAINT                         |
+-------------------------------+-------------------------------+
|            Status             |  MaintOpcode = Read Errorlog  |
+-------------------------------+-------------------------+-----+
|                         Error Log Data                        |
|                                                               |
+---------------------------------------------------------+-----+
```

2.4.1.1.10.4 Disable Error Log Command

Macro diagnostics may will perform tests which will cause errors on the Adapter. This command allows them to disable error logging for those expected error cases. They must ensure that they then reenable error logging upon completion of their testing.

```
 31                             ...                             0
+---------------------------------------------------------------+
|                        Opcode = MAINT                         |
+-------------------------------+-------------------------------+
|            Status             | MaintOpcode = Disable Errorlog |
+-------------------------------+-------------------------+-----+
```

2.4.1.1.10.5 Enable Error Log Command

Macro diagnostics may will perform tests which will cause errors on the Adapter. They will first disable error logging to prevent unnecessary writes to the Adapter Error Log. At the end of testing they must then reenable error logging. The default is for the error logging to be enabled across Adapter resets.

```
 31                                ...                              0
+-------------------------------------------------------------------+
|                           Opcode = MAINT                          |
+-------------------------------+-----------------------------------+
|             Status            | MaintOpcode = Enable Errorlog     |
+-------------------------------+----------------------------+------+
```

2.4.1.1.10.6 EEPROM Update Commands

When the Adapter is in the Initialized or Running state, the Host may issue maintenance commands to cause the firmware to update the contents of EEPROM on the module. EEPROM Update Commmands received in adapter states other than Initialized or Running result in an Illegal State for Command error. See the Updating Firmware chapter for details.

Note: If an EEPROM Update Command that would cause a write to EEPROM is received by the Adapter Manager when the XMI UPDATE EN line is not asserted, the Adapter Manager will not process the command and will return an "Invalid Command" error. The state of XMI UPDATE EN is provided in the ESP Gate Array CSR interface.

2.4.1.1.10.6.1 EE$ReadParam Command

This command causes the Adapter Manager to write the the operational adapter parameters which are stored in EEPROM to the Command Buffer.

Figure 2–13: EE$ReadParam Command Format

```
 31                                ...                              0
+-------------------------------------------------------------------+
|                           Opcode = MAINT                          |
+-------------------------------+-----------------------------------+
|             Status            |   MaintOpcode = EE$ReadParam      |
+-------------------------------------------------------------+-----+
|                         RSVD                                | BOO |
+-------------------------------------------------------------+-----+
|                  Firmware image revision date                     |
|                     (12 ASCII Characters)                         |
|                  (for example, '12-JUL-1988 ')                    |
+-------------------------------------------------------------------+
|                     Firmware image revision                       |
|                      (4 ASCII Characters)                         |
|                      (for example, '0200')                        |
+-------------------------------------------------------------------+
```

2.4.1.1.10.6.2 EE$WriteEEPROM Command

This command causes the Adapter Manager to copy the EEPROM image data from host memory to adapter EEPROM.

Figure 2–14: EE$WriteEEPROM Command Format

```
 31                           ...                            0
+--------------------------------------------------------------+
|                        Opcode = MAINT                        |
+------------------------------+-------------------------------+
|            Status            |   MaintOpcode = EE$WriteEEProm |
+------------------------------+-------------------------------+
|        Longword Length       |     Lword Offset (0..64K-1)   |
+------------------------------+-------------------------------+
|              Physical Address of Code Segment (31:0)         |
|                    (physically contiguous)                   |
+------------------------------------------------+           --+
                                                 |   (39:32)    |
                                                 +--------------+
```

Where:

- Longword Length: The length of the image segment in longwords.
- Lword Offset: The longword offset of the segment within the EEPROM image.
- Physical Address of Code Segment: Address of the code segment in Host memory.

2.4.1.1.10.6.3 EE$WriteLast Command

This command is issued by the Host when the image being provided is the last segment of a complete EEPROM image update. This command causes the Adapter Manager to copy the EEPROM image data from host memory to adapter EEPROM, and do a CRC calculation on the entire image in EEPROM. The result is compared with the CRC in the last 4 bytes of the image segment provided by this command. If the CRCs match, the command succeeds. If the CRCs do not, the command fails. If the command fails, the Host has the option of restarting the update process. If the command succeeds, the Host may reset the Adapter, causing the new image in EEPROM to be executed.

Figure 2–15: EE$WriteLast Command Format

```
 31                           ...                            0
+--------------------------------------------------------------+
|                        Opcode = MAINT                        |
+------------------------------+-------------------------------+
|            Status            |   MaintOpcode = EE$Write_Last |
+------------------------------+-------------------------------+
|        Longword Length       |     Lword Offset (0..64K-1)   |
+------------------------------+-------------------------------+
|              Physical Address of Code Segment (31:0)         |
|                    (physically contiguous)                   |
+------------------------------------------------+           --+
                                                 |   (39:32)    |
                                                 +--------------+
```

2.4.1.1.10.6.4 EE$ReadEEPROM Command

This command causes the Adapter Manager to copy the EEPROM image data from adapter EEPROM to host memory.

Figure 2–16: EE$ReadEEPROM Command Format

```
 31                              ...                              0
+-----------------------------------------------------------------+
|                         Opcode = MAINT                          |
+--------------------------------+--------------------------------+
|             Status             |  MaintOpcode = EE$ReadEEProm   |
+--------------------------------+--------------------------------+
|         Longword Length        |    Lword Offset (0..64K-1)     |
+--------------------------------+--------------------------------+
|              Physical address of 64k image (31:0)               |
|                    (physically contiguous)                      |
+-------------------------------------------------+             --+
                                                  |   (39:32)     |
                                                  +---------------+
```

2.4.1.1.10.6.5 Read$Boot Command

This command causes the Adapter Manager to copy the most recently received MOP Boot message (if one has been received) since the last Read$Boot command from adapter EEPROM. This command is useful to a console boot driver which has just been invoked after the Boot message caused an XMI Reset. The console boot driver, if it is going to perform a remote boot, needs the contents of the Boot message to determine what image to ask for in the MOP Request Program message it will send to initiate the boot sequence.

If the command buffer supplied is not long enough to hold the entire Boot message, command ring entry error code and the maintenance command status will be 'Buffer length error'.

Receipt of Read$Boot causes the Adapter Manager to invalidate the Boot frame currently in EEPROM.

Figure 2–17: Read$Boot Command Format

```
 31                              ...                              0
+-----------------------------------------------------------------+
|                         Opcode = MAINT                          |
+--------------------------------+--------------------------------+
|             Status             |     MaintOpcode = Read$Boot    |
+--------------------------------+--------------------------------+
|                                                                 |
:                        MOP Boot Message                         :
|                                                                 |
+-----------------------------------------------------------------+
```

2.4.2 Transmit Ring

The transmit ring is a physically contiguous data structure located in host memory aligned to a 512-byte boundary. It consists of a number of quadword-sized ring entries. The number of entries in the transmit ring is set at initialization time, based on the parameters contained in the PDB.

The maximum number of transmit entries is limited by the addressing range of the host interface logic. Refer to Appendix D for specific information.

A single transmit frame is contained in one or more transmit entry buffers.

Each ring entry gives status and buffer segment address information and provides a place to put status information when processing of the ring entry by the adapter is complete.

Transmit ring entries are only processed in the *Running* or *Maintenance* states.

The buffer described by the ring entry contains a transmit buffer. A transmit buffer contains a frame or a portion of a frame to be transmitted on the FDDI.

The buffer pointed to by a transmit entry consists of a single buffer segment. If the frame size exceeds the buffer size of the initial entry, then succeeding entries' buffers are used to contain the remainder of the frame, i.e. buffer chaining.

The EOP and SOP bits described below control buffer chaining. The transmit entry pointing to the buffer containing the start of the FDDI frame will have SOP set. The transmit entry pointing to the buffer containing the end of the FDDI frame will have EOP set. If both SOP and EOP are set in a single entry, that entry contains the entire frame.

The first transmit entry looks like:

Figure 2–18: Initial Transmit Ring Entry

```
  31  30  29  28...............16  15 14............9 8..............0
+---+---+---+-------------------+---+-------------+---------------+
|Own|SOP|EOP| Segment Length    |ERS|     RSVD    |  PA_LO (8:0)  |
+---+---+---+-------------------+---+-------------+---------------+
| X |             Buffer Segment Physical Address   (39:9)         |
+---+------------------------------------------------------------+
```

Where:

- <31> Status/Own. (See Command Ring Entry)
- <30> SOP. Indicates that the buffer associated with this entry contains the "Start of Frame".
- <29> EOP. Indicates that the buffer associated with this entry contains the "End of Frame".
- <28:16> Segment Length. This field gives the Buffer Segment length in bytes.
- <15> ERS. (See Command Ring Entry)
- <8:0> PA_LO. Lower 9 bits of physical address of buffer segment to transmit.
- <31> NIO
- <30:0> Buffer Segment Address. High 31 bits of physical address of buffer segment to transmit.

If the first transmit entry buffer is insufficient to contain the command, subsequent entries are allocated as follows:

Figure 2–19: Transmit Ring Entry for Additional Segments

```
  31  30  29  28...............16 15..............9 8.............0
+---+---+---+-------------------+-----------------+---------------+
|RSV|SOP|EOP| Segment Length    |       RSVD      |  PA_LO (8:0)  |
+---+---+---+-------------------+-----------------+---------------+
| X |            Buffer Segment Physical Address   (39:9)          |
+---+-------------------------------------------------------------+
```

Note: When the driver processes the return entries for all packets transmitted by the adapter it must write the address of the last packet transmit entry it worked on to the TRANSMIT HIBERNATION registers to notify the adapter it is hibernating. In the case where a packet required multiple transmit entries the driver must write the address of the last entry that contained the 'OWN' bit.

2.4.2.1 Transmit Buffer Format

A transmit buffer is simply a buffer containing a transmit frame or a portion of a transmit frame. Each entry buffer may begin and end on any byte boundary. The initial entry contains the entire FDDI header portion of the frame, including the first longword, which is used for FDDI header data provided by the port driver. The transmit buffer for a single entry frame looks like:

Figure 2–20: Transmit Buffer Format

```
 31                                                          0
+------------+-----------------------------------------------+
|Frame Control|         Reserved for Frame  Header            |
+------------+-----------------------------------------------+
|                      Destination Address                    |
+-----------------------------+                            --+
|                             |                               |
+                             +-------------------------------+
|                        Source Address                       |
+-------------------------------------------------------------+
|                    802 Header Data (3 - 8 bytes)            |
|                    (If this in an LLC frame )               |
+-------------------------------------------------------------+
|                           User Data                         |
|                      (0 - Max Frame  bytes)                 |
+-------------------------------------------------------------+
|                        FDDI CRC Field                       |
+-------------------------------------------------------------+
```

All fields are supplied by the port driver with the exception of the CRC. Subsequent buffers, if allocated, contain user data exclusively. See Appendix F for a description of the default packet header.

2.4.3 Receive Ring

This is a physically contiguous ring of receive entries aligned to a 512-byte boundary. The end of the receive ring must coincide with the end of a receive entry.

The maximum number of receive entries is limited by the addressing range of the host interface logic. Refer to Appendix D for specific information.

Each receive entry consists of a set of pointers to one or more buffers used as a repository for a received FDDI frame.

Receive ring entries are only processed in the *Running* or *Maintenance* states.

The size of a single receive entry is an integral number of octawords. For a given instantiation, all receive entries are the same size. The smallest entry is a single octaword, as shown.

Figure 2–21: Receive Ring Format

```
  31  30  29 28.....................16 15 14 13......9 8.............0
+---+---+---+----------------------+--+--+---------+--------------+
|Own|SOP|EOP| Segment Leng/Byte Coun | M| U|  Uindex | PA_LO (8:0)  |
+---+---+---+----------------------+--+--+---------+--------------+
|LST|           Buffer Segment Physical Address   (39:9)           |
+---+---+---+----------------------+-----+------------------------+
|ERS|NIO|FSC|     Segment Length     |FSC  |RMC FSB  | RMC RCC       |
+---+---+---+----------------------+-----+---------+--------------+
|LST|           Buffer Segment Physical Address High (39:9)        |
+---+-------------------------------------------------------------+
```

Where:

- First Quadword:
  - <31> Status/Own. Ownership bit for this ring entry. If set, the entry is owned by the host. If clear, owned by the adapter. The ownership bit is cleared by the host after it has set up the ring entry.

    A ring entry not owned by the host means that no field other than the ownership bit may be considered valid by the host. A ring entry not owned by the adapter means that no field other than the ownership bit may be considered valid by the adapter.
  - <30> SOP. Start of frame bit. Used for the chaining of several entries. The adapter will set this bit to denote the first entry required for the current receive buffer. The adapter will clear this bit for all intermediate entries required for the current receive buffer.
  - <29> EOP. End of frame bit. Used for the chaining of several entries. The adapter will set this bit to denote the last entry required for the current receive buffer. The adapter will clear this bit for all intermediate entries required for the current receive buffer.
  - <28:16> Segment Length/Total Byte Count. Initially, when read by the adapter, this field contains the length, in bytes, of the buffer segment pointed to by the physical address contained in the first quadword. This length must be an integral number of hexawords. When written by the adapter, this is the length of the receive frame contained in the buffer(s) pointed to by the entry buffer segment addresses. When

the adapter copies a receive frame into the buffer, it sets the receive length to the actual length of the received frame, including header, data, and CRC.

- <15> M. When set, indicates that the received frame should be delivered to multiple users. This bit will always be set for any frame which contains a GSAP. The Port Driver is expected to do further filtering to determine which if any other users are to receive this frame.
- <14> U. When set, indicates that the received frame is intended for the unknown user.
- <13:9> User_Index. User Definition Block index. This is the user number assigned to the user by a previous USTART command, numbered 0..31. It identifies the user who is to receive the frame. This field is not valid if the parser forwarding vector has the M bit set. The driver must perform full filtering on any packet with the M bit set.
- Buffer Segment Address. This is the physical address of the buffer. It must start on a hexaword aligned address. The ending address of the buffer should also be hexaword aligned.
- <31> LST. The last segment bit. If set, it notifies the adapter that the current segment is the last valid segment for the current entry. If clear, the adapter knows that there are more segments available for the current entry.

- Second Quadword:
  - ERS. Error summary field for the received frame. If set, the error field gives the reason for failure. If clear, the error code is zero.
  - FSC. Frame Status Count bit 29 from RMC RCV buffer descriptor. See RMC specification for further details.
  - Segment Length. This field defines the number of valid bytes associated with the segment address. It is read-only for the adapter. This must be an integral number of hexawords.
  - FSC. Frame Status Count bits 27 and 28 from RMC RCV buffer descriptor. See RMC specification for further details.
  - RMC FSB. Frame Status Bits 22:26 from RMC RCV buffer descriptor. See RMC specification for further details.
  - RMC RCC. Receive Completion Code from RMC RCV buffer descriptor. See RMC specification for further details.
  - LST. The last segment bit. If set, it notifies the adapter that the current segment is the last valid segment for the current entry. If clear, the adapter knows that there are more segments available for the current entry.
  - Buffer Segment Address. This field defines the high part of the physical address required to store data associated with this segment. The lower nine bits of the buffer segment address are MBZ, i.e. it is 512-byte aligned.

If the receive entry size is greater than one octaword, the format of the third and succeeding quadwords repeat that of the second quadword; the only difference is the second quadword of an entry contains the error bits (ERS, FSC and RMC FSB, RMC RCC), while succeding quadwords do not.

If there are succeeding octawords for a receive entry their format is:

Figure 2–22: Receive Ring Entry for Second and Succeeding Octawords

```
  31......29 28...................16 15...........................0
+-----------+-----------------------+-----------------------------+
| Reserved  |     Segment Length    |          Reserved           |
+---+-------+-----------------------+-----------------------------+
|LST|           Buffer Segment Physical Address   (39:9)          |
+---+-------+-----------------------+-----------------------------+
| Reserved  |     Segment Length    |          Reserved           |
+---+-------+-----------------------+-----------------------------+
|LST|           Buffer Segment Physical Address High (39:9)       |
+---+-------------------------------------------------------------+
```

Note: When the driver is done processing receive entries for packets delivered to the host it must write the address of the last receive ring entry it worked on to the RECEIVE HIBERNATION register. In the case where chaining occurs, the driver must write the address of the last entry of the chained buffer to the RECEIVE HIBERNATION register.

2.4.3.1 Receive Buffer Format

The following section describes the format of the receive buffer. A receive buffer is simply a buffer containing either an FDDI receive frame or a portion of a frame. Each portion of a receive frame buffer segment must begin and end on a hexaword-aligned address. This is needed to simplify chaining, saving costly masking for the sub-hexaword fragments. The second and succeeding buffers for a receive entry must start on a 512 byte aligned boundary. The receive buffer format for a single buffer segment frame has the following form:

Figure 2–23: Receive Buffer Format

```
 31        24  23                                           0
+-------------+---------------------------------------------+
|Frame Control|         Reserved for Frame  Header          |
+-------------+---------------------------------------------+
|                     Destination Address                   |
+-----------------------------+                           --+
|                             |                             |
+--                           +-----------------------------+
|                       Source Address                      |
+-----------------------------------------------------------+
|                 802 Header Data (3 - 8 bytes)             |
|                  (If this in an LLC frame )               |
+-----------------------------------------------------------+
|                         User Data                         |
|                (0 - Max Receive Frame )                   |
+-----------------------------------------------------------+
|                      FDDI CRC Field                       |
+-----------------------------------------------------------+
```

Subsequent buffers, if allocated, contain user data exclusively.

2.4.4 Unsolicited Ring

This is a physically contiguous ring of unsolicited entries aligned to a 512-byte boundary. An unsolicited entry consists of a single hexaword used for unsolicited adapter-to-host communication. For events which the adapter needs to indicate to the host, the unsolicited ring serves as a medium for signalling them.

Figure 2–24: Unsolicited Ring Format

```
 31                               16 15                           0
+---+-------------------------------+-----------------------------+
|Own|         Reserved              |     Unsolicited Opcode      | A
+---+-------------------------------+-----------------------------+
|                       Unsolicited Data #1                        | A+4
+-----------------------------------------------------------------+
|                       Unsolicited Data #2                        | A+8
+-----------------------------------------------------------------+
|                       Unsolicited Data #3                        | A+C
+-----------------------------------------------------------------+
|                       Unsolicited Data #4                        | A+10
+-----------------------------------------------------------------+
|                       Unsolicited Data #5                        | A+14
+-----------------------------------------------------------------+
|                       Unsolicited Data #6                        | A+18
+-----------------------------------------------------------------+
|                       Unsolicited Data #7                        | A+1C
+-----------------------------------------------------------------+
```

Where:

- <31> Own. Ownership bit for this ring entry. If set, the entry is owned by the host. If clear, owned by the adapter. The ownership bit is cleared by the host after it has set up the ring entry.

  A ring entry not owned by the host means that no field other than the ownership bit may be considered valid by the host. A ring entry not owned by the adapter means that no field other than the ownership bit may be considered valid by the adapter.
- <15:0> Unsolicited Opcode. See Appendix A for the values of unsolicited opcodes.
- Unsolicited Data, seven longwords of indication specific information.
  - Unsolicited Data, seven longwords. In the case of the DIRECTED BEACON RECEIVED message, the first 28 bytes of the DIRECTED BEACON frame are saved in these locations. For all other unsolicited messages no further information is needed in these locations.

The size of a single unsolicited entry is one hexaword.

2.4.4.1 Unsolicited Entry Types

The following section describes the types of the unsolicited entries. An unsolicited entry contains an adapter-initiated indication for the host. See Appendix A for the values of unsolicited opcodes.

The following events are reported to the host through the UNSOLICITED ring. See CNS documentation for a complete description of each.

- RING INIT INITIATED

- RING INIT RECEIVED
- RING BEACON INITIATED
- DUPLICATE ADDRESS DETECTED
- DUPLICATE TOKEN RECEIVED
- RING PURGE ERROR
- BRIDGE STRIP ERROR
- RING OP OSCILLATION
- DIRECTED BEACON RECEIVED
- PC TRACE INITIATED
- PC TRACE RECEIVED
- TRANSMIT UNDERRUNS
- TRANSMIT FAILURES
- RECEIVE OVERRUNS
- LEM REJECT
- EBUFF ERROR
- LCT REJECT

CHAPTER 3

PORT STATES

3.1 Adapter States

The XFA Adapter has six states: Resetting, Uninitialized, Initialized, Running, Halted, and Maintenance. Any state transition except from Resetting to Uninitialized is reported by a Host interrupt. No interrupt is generated between Resetting and Uninitialized because the Host interrupt vectors are not known at that point.

The following sections contain descriptions of the XFA States as well as what events cause transition into them. The list of Host accessible functions (i.e. Command Processing, receipt of FDDI Frames) for each state is also included.

3.1.1 Resetting

The Resetting State indicates the Adapter is resetting itself, and will shortly transition to the Uninitialized State, unless an error occurs. Resetting includes loading and running Self Test. If an error occurs during the Resetting State, the adapter will not transition to the Uninitialized State (it stays in the Resetting State). The Self Test Fail (STF) bit in the XBE register will not be clear. In this case the device driver will timeout and return Fatal Controller Error.

The adapter enters the Resetting State in the following cases:

- Power Up.
- XMI Node Reset.

The following functions are available to the Host when the XFA is in the Resetting State:

- None. The adapter is in transition, and may be performing Self Test. No ring services are available.

3.1.2 Uninitialized

The Uninitialized State indicates the Adapter has run diagnostics successfully, RTOS has been booted, hardware and packet memory have been intialized, and the adapter is ready to complete initialization by the Host through a write of the XPCI register.

The adapter enters the Uninitialized State in the following cases:

- At Power Up, following successful completion of Self Test and successful RTOS boot, the adapter will transition from the Resetting State to the Uninitialized State.
- From the Initialized, Halted, Running, and Maintenance States when the Host writes the XPCS (Port Shutdown) register.

The following functions are available to the Host when the XFA is in the Uninitialized State:

- INIT command processing (write of XPCI)
- XMI Node Reset processing (write of XBE bit 30)
- No ring services are available.

3.1.3 Initialized

The Initialized State indicates the adapter has been initialized by the host. The Adapter Manager has successfully processed data in the Port Data Block, allocated all buffers for internal memory, loaded all registers with addresses of ring structures in host memory, completed the ESP Special Test, and is finally ready to process Commands on the Host Command Ring.

The adapter enters the Initialized State in the following cases:

- From the Uninitialized State, following INIT command (write of XPCI) by the host.

  Following receipt of the Init command, before transitioning to the Initialized State, the Adapter performs a Test of the ESP functions by utilizing an internal loopback mechanism to loop a packet from the Host Transmit Ring through the adapter and back to the Host Receive Ring.

  If an error occurs during the transition between Uninitialized and Initialized states, the Adapter remains in the Uninitialized State, and places error information in the Port Status and Port Data Registers (see Appendix A). The device driver will timeout waiting for the initialization to complete and report the event as a Fatal Controller Error.

The following functions are available to the Host when the XFA is in the Initialized State:

- XMI Halt command processing (write of XBE bit 29)
- XMI Node Reset processing (write of XBE bit 30)
- Shutdown command processing (write of XPCS)
- Host Command Ring processing. EEPROM Update commands are accepted in this state.

3.1.4 Running

The Running state indicates the adapter has been initialized by the Host. A PARAM command has been received and successfully completed. The ring quality tests have completed successfully, and the FDDI connection 'request' has been successfully received. The device driver is then able to start accepting requests from users to transmit packets and queue those packets to the adapter for transmission. It may be that at this time, packets cannot be physically transmitted since it is some time after the request for the connection that the actual connection is established. Once the connection is established however, the packets can be sent and received to and from the fiber. See the section 3.3 of this document for relationship between FDDI states and Adapter states.

The adapter enters the Running State in the following cases:

- From the Initialized State following successful completion of a PARAM command received from the Host. The PARAM command signals the adapter to attempt to connect to the FDDI Ring with the FDDI parameters provided. See PARAM Command definition in Chapter 2.

The following functions are available to the Host when the XFA is in the Running State:

- XMI Halt command processing (write of XBE bit 29)
- XMI Node Reset processing (write of XBE bit 30)
- Shutdown command processing (write of XPCS)
- Host Ring Processing for all Host rings (Transmit, Command, Receive, and Unsolicited). EEPROM Update commands are accepted in the Running state.

3.1.5 Maintenance

The adapter enters the Maintenance State in the following cases:

- From the Initialized State following successful processing of a PARAM command with LM = 1 (External Loopback), or LM = 2(CDC Loopback).

The following functions are available to the Host when the XFA is in the Maintenance State:

- XMI Halt command processing (write of XBE bit 29)
- XMI Node Reset processing (write of XBE bit 30)
- Shutdown command processing (write of XPCS)

- Host Ring Processing for all Host rings (Transmit, Command, Receive, and Unsolicited). EEPROM Update commands are accepted in the Maintenance state.

3.1.6 Halted

The adapter enters the Halted State in the following cases:

- From the Uninitialized, Initialized, Running, or Maintenance States when the XMI Node Halt bit is set by the host.
- From the Uninitialized, Initialized, Running, or Maintenance States when an internal error is encountered by the adapter.

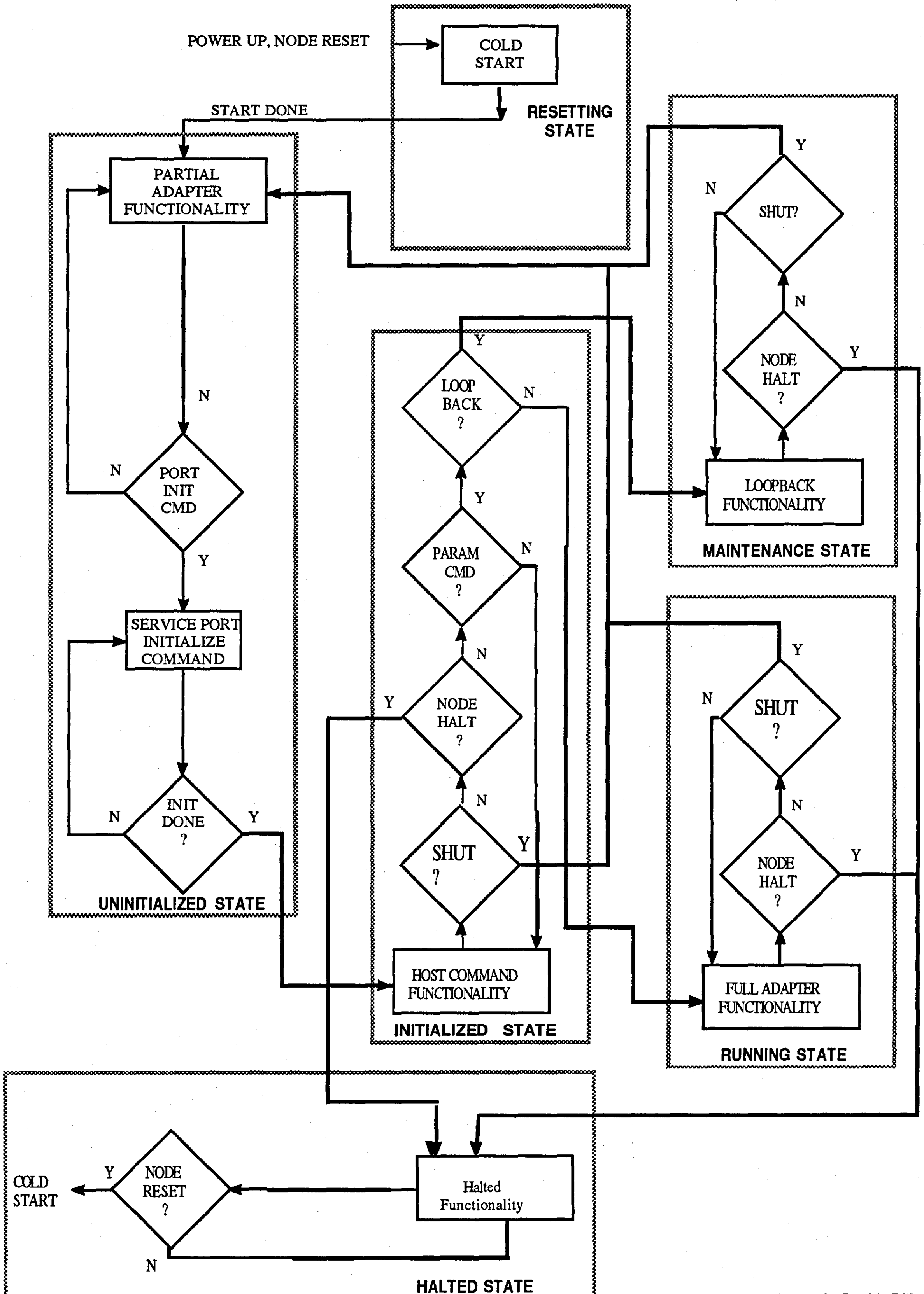
The following functions are available to the Host when the XFA is in the Halted State:

- XMI Node reset (setting bit 30 in XBE) or Power Up reset.

The Adapter only exits the Halted State through the host issuing XMI Node Reset (setting bit 30 in the XBE register).

The relationship among the six port states is shown in the following diagram:

Figure 3–1: XFA PORT STATE DIAGRAM



3.2 FDDI States

Although the FDDI states will remain transparent to the device driver we do list them here to provide an understanding of the FDDI ring's AVAILABILITY. Rather than complicate the Port Driver with all the ring states, we simplified the drivers world by pairing the adapter states with the AVAILABILITY/UNAVAILABILITY of the ring. See the XPST register definition in Chapter 2 for further details. The host is interrupted on Adapter state changes and ring availability changes. For a more detailed description, see the FDDI Data Link Architecture. Following each FDDI state below is a list of the XFA Port States in which that FDDI Ring State may occur.

3.2.1 Off Init

In this state, the XFA is not attached to the FDDI Ring, and no data is transmitted or received. During this state, Self Test may be executing. The FDDI ring is UNAVAILABLE to the host. This FDDI State may occur during the following XFA Port States:

- Resetting

3.2.2 Off Ready

This FDDI state indicates the FDDI chips have passed self test, and are in working order. The XFA is not attached to the FDDI Ring, no data is transmitted or received. The ring is UNAVAILABLE. In this state the Adapter Manager may modify FDDI characteristics through CNS. This FDDI State may occur during the following XFA Port States:

- Uninitialized
- Maintenance
- Halted
- Initialized

3.2.3 On Ring Init

This FDDI state indicates a physical connection to the FDDI ring has been established. In this state the MAC is performing its initialization process where claim frames are generated in the attempt to synchronize the ring by generation of a single non-restricted token. Although the ring is unavailable for physical transmissions the host may accept frames for transmission and forward them to the adapter. The FDDI ring is considered AVAILABLE to the host. This FDDI State occurs in the following XFA States:

- Running

3.2.4 On Ring Run

In this FDDI state, physical and logical connections to the FDDI Ring have been established, and the ring is AVAILABLE. This FDDI State may occur in the following XFA States:

- Running

3.2.5 Off Fault Recovery

This is the fault recovery state where Common Node Software has detected an event which requires special fault handling. An example of this is when CNS detects the absense of any token for more than 2 seconds. CNS must then start the PC TRACE algorithm to attempt recovery from the faulting condition. While this activity of going on the FDDI state is in OFF_FAULT_RECOVERY and the ring is UNAVAILABLE to the host. For further details of this state see CNS documentation. This FDDI State occurs in the following XFA States:

- Running

Note that if the adapter stays in this state for TBD seconds, an adapter reset will occur only after the problem has been resolved or the system manager has explicitly reset the adapter. This is necessary to flush the transmit queues of stale packets.

3.2.6 Broken

This State indicates a fatal error has been detected by Self Test which will prevent proper operation on the FDDI. The ring is UNAVAILABLE. This state may occur in the following XFA States:

- Resetting. If there are any failures in Self Test, the Adapter will remain in the Resetting State.

3.3 Relationship Between XFA Port States and FDDI Ring State Transitions

3.3.1 Power Up State Transitions

The following events occur during power up:

1. Self Test executes. Port State is Resetting, XFA FDDI State is Off Init.
2. If Self Test fails, FDDI State may go to Broken, and Port State will remain Resetting.
3. If Self Test completes successfully then RTOS boots, a partial init completes, the FDDI State goes from Off Init to Off Ready, the Port State goes to Uninitialized, and STF gets cleared.
4. The Host issues the Init command.
5. The Adapter Manager reads the Port Data Block, and completes initialization of the Adapter Hardware and the Firmware Data Structures. Diagnostics code will execute a test of the ESP Functions that use internal loopback of a packet from the Host Transmit Ring through the adapter back to the Host Receive Ring.
6. The Port State goes to Initialized.

7. The Host issues a PARAM command.
8. The Adapter Manager reads the parameters in the PARAM command.
9. The Adapter Manager requests CNS to connect to the ring.
10. The Adapter Manager changes the Adapter State to Running, and then returns success status for the PARAM command.
11. The FDDI state transitions to the On Ring Init state in an attempt to connect to the ring. If the connection attempt is eventually successful, the Adapter FDDI state goes to On Ring Run. XPST is written notifying the host that the ring is AVAILABLE.

3.3.2 FDDI State Transitions During XFA Running State

- When a new node is added to or removed from the ring, the logical connection will be broken and the FDDI State transitions through On Ring Init back to On Ring Run. The FDDI ring is still considered available to the host processor during this interval.

3.3.3 State Transitions Following Errors

The following describes the State transitions that result from fatal errors. See also Chapter 6, "Adapter Manager Internal Event Handling" for a list of fatal errors.

Error case of:

- Watchdog Timer Expires: No state change, WTO in XPST gets asserted, Host interrupted, Adapter Manager vectored to ISR that remains at the highest IPL level awaiting node reset.
- ESP Bus Broken Error: No state change, AMPE bit in XPST gets asserted, Host interrupted, Adapter Manager vectored to ISR that remains at the highest IPL level awaiting the node reset.
- SRAM Parity Error: No state change, SPE bit in XPST gets asserted, Host interrupted, Adapter Manager frozen. Hardware generates a node reset.
- All other Adapter Manager/CNS detected errors cause the Adapter to enter the HALTed state and stop processing either packets from the FDDI ring or Host Command entries. If error logging is enabled then an internal EEPROM Error Log is also written with internal Adapter registers to capture the error signature. Transitioning to the HALTed state causes a host interrupt. Transitioning out of the HALTed state requires a node reset by the host.

3.3.4 State Transitions Initiated By The Host

The Host may cause state transitions through the following mechanisms:

- Init Command (write of the XPCI register): Valid only in the Uninitialized State. Adapter Manager reads the Port Data Block and caches it. When the adapter is ready to process commands from the Command Ring, Adapter State goes to Initialized. FDDI State remains Off Ready until the Host issues a PARAM command.
- PARAM Command (through Host Command Ring, valid only in Initialized State): Following completion of the PARAM Command, the Adapter State is Running.

- Shutdown Command (XPCS register write): Valid in the Maintenance, Initialized or Running States. Causes the Adapter Manager to do internal cleanup of all internal data structures, and transition to the Uninitialized State. FDDI State goes to Off Ready.
- XMI Node Reset (write of bit 30 in the XBE): Valid in all Port States. The adapter performs a power up reset. No error log written. Port State goes to Resetting, and FDDI State goes to Off Init. If Self Test completes successfully, the Port state will go to Uninitialized and the FDDI State to On Ring Init.
- XMI Node Halt (write of bit 29 in the XBE): Valid in any state except Resetting or Uninitialized. Causes the Adapter to save as much internal state as possible and disables processing of received FDDI packets and Host Command Ring entries. Port State goes to Halted. FDDI State goes to Off Ready.

3.3.5 PORT State Processing

The port state determines how the adapter responds to external influence, both from the host or port driver and from other nodes on the FDDI.

With regard to the host, the adapter accesses host memory when the port state is *Running*, *Maintenance*, *Initialized*, or *Halted*. It also accesses host memory during the servicing of a Port Init State Command (while the port is *Uninitialized*).

With regard to the FDDI, the adapter responds as follows to frames received, either addressed to the specified address or for periodic System ID messages:

Table 3–1: Port State vs FDDI Activity

State vs. Frame Type	MOP and SMT	Periodic System ID	Loopback	802 Test/XID	User Frames
Resetting	No	No	No	No	No
Uninitialized	No	No	No	No	No
Initialized	No	No	No	No	No
Running	MLA	MLA	MLA	EA	EA
Maintenance	No	No	No	No	No
Halted	No	No	No	No	No

Where:

- MLA - Physical address stored in the Default Physical Address ROM.
- EA - All enabled physical addresses for this set of frames. One of these is the MLA. The remainder are enabled by users.

With regard to host ring transactions, the adapter responds as follows to entries queued to each type of host ring:

Table 3–2: Port State vs Ring Processing

State vs. Ring Entry Type	Command	Unsolicited	Transmit	Receive
Resetting	No	No	No	No
Uninitialized	No	No	No	No
Initialized	Yes	No	No	No
Running	Yes	Yes	Yes	Yes
Maintenance	Yes	Yes	Yes	Yes
Halted	No	No	No	No

The relationship between the Port State and the FDDI Link State is shown in the table below:

Table 3–3: Port State and allowed FDDI States

Port vs. FDDI State	Off Init	Off Ready	On Ring Init	On Ring Run	Fault Recovery
Resetting	Yes	No	No	No	No
Uninitialized	No	Yes	No	No	No
Initialized	No	Yes	No	No	No
Running	No	No	Yes	Yes	Yes
Maintenance	No	No	Yes	Yes	Yes
Halted	No	Yes	No	No	No

CHAPTER 4

PORT INITIALIZATION

4.1 Port Power Up

On power up, the firmware performs the following sequence to INIT the Adapter:

- Run self test and write self test results to the Power Up Diagnostic Register.
- Boot RTOS by scheduling RTOS_Init task.
- Load the MLA, into the MAC Chip
- Enables 68020 internal interrupts.
- If everything executed successfully clear *Self Test Fail* bit in the XMI Bus Error Register.
- Writes port state *Uninitialized* to the XPST Register (with state qualifier 'No Error').

If self test fails but the module is sufficiently workable to load firmware and possibly run diagnostics, the *Self Test Fail* bit is not cleared from the XMI Bus Error Register and the state qualifier 'Port failed self test' is written to the XPST Register.

4.2 Port Driver Initialization Sequence

The port driver performs the following sequence to INIT the adapter:

For the simple case, where the adapter has just been powered up, the steps are:

1. Wait for self test to complete, waiting up to 10 seconds for *Self Test Fail* bit to clear from the XBE Register
2. If the STF bit is not clear, return failure
3. Read the XDEV Register and verify that this is a XFA module.
4. Initialize the port driver data structures, Port Data Block, Transmit, Unsolicited, Command and Receive Rings and any other associated data structures needed.
5. Write the physical address of the Port Data Block to the XPD1 and XPD2 Registers.
6. Issue an *INIT* command, by writing the XPCI Register.
7. As part of the INIT Command, the adapter will conduct testing by creating self-directed entries on the host transmit ring, and initiating a host interface test which will loop these frames back to the host receive ring. The port driver is expected to provide 1 transmit buffer and 2 receive buffers accomodate this ESP special test.
8. Wait for initialization to complete, waiting up to 2 seconds for the status interrupt from the adapter indicating a state transition to the *Initialized* state.
9. If the *Initialized* state appeared in the XPST Register, return success.

10. If the *Init* command failed, check the state qualifier to determine why - if the error code is 'Port initialization failed, adapter failed self test', return failure status to the user who requested startup.
11. Once the adapter is in the *Initialized* state: issue a PARAM command to enable the FDDI link.
12. Once the PARAM command entry is returned with success status, the driver may issue USTART commands immediately.

4.3 Adapter Receives INIT Command

- In the Uninitialized State wait for INIT register write from the host.
- Service the XPCI Interrupt, which indicates an *Init* command.
- Read and validate the Port Data Block address in the XPD1 and XPD2 Registers.
- Read and validate the Port Data Block contents.
- Initialize appropriate adapter firmware data structures based on the data contained in the Port Data Block.
- Initialize PMC data structures.
- Execute host interface logic testing. This involves setting up loopback, creating self-directed entries on the host transmit ring, and initiating a host interface test which will loop these frames back to the host receive ring.
- Set internal port state to *Initialized* and update the XPST Register.
- Once reaching the *Initialized* state, the adapter processes commands from the host. It remains in this state unless it completes a PARAM command, a SHUTDOWN command is issued, or an error occurs.

4.4 Link Enable

- Once the adapter completes a PARAM command it will transition from the *Initialized* state to either the *Running* or the *Maintenance* state.
- If the PARAM command does not indicate a loopback mode, the following steps are completed:
  - Enable FDDI interrupts.
  - Initializes FDDI Chipset using parameters specified in the PARAM command.
  - Request a logical and physical connection to network
  - Sets Port state to *Running*.

  Note: It may be some time later after the request to establish a connection, that the actual connection is established. The adapter does not wait for the actual connection to be made. The state of the adapter is changed to RUNNING. The driver may start users immediately. However no frames will be sent or received until the connection is finally established and the ring is AVAILABLE. Availability is reported to the driver via the XPST register.

- If the PARAM command indicates a loopback mode, the following steps are completed:
  - Enable FDDI interrupts.
  - Initializes FDDI Chipset using parameters specified in the PARAM command.
  - Request a logical and physical connection to network
  - Sets Port state to *Maintenance*.

CHAPTER 5

PORT SHUTDOWN

5.1 Adapter Shutdown

Adapter shutdown consists of the following actions done by operational firmware:

- Disengage from the ring using SMT procedures.
- Cancel all scheduled RTOS tasks.
- Internal flags that have to do with initialized operation are reset. Fields in the adapter internal copy of the Port Data Block are set to their default values.
- Save XPUD register
- Generate an AMI quick reset, in which firmware continues to run.
- Reboot RTOS
- Restore XPUD register
- The address of the host copy of the port data block is reset. The new value will be provided by the host when it brings the adapter back up to *Initialized.*
- The host memory ring context, i.e. ring locations, next pointers and lengths, are reset. When the adapter is reinitialized by the host, new host ring context will be obtained from the host supplied fields in the PDB.
- Reset the Watchdog Timer.
- Change state to *Uninitialized.*

5.2 Power Fail Shutdown

On detection of power fail, the adapter gets interrupted at IPL7, loads XPST with HALTed state with a state qualifier specifying a Power Fail reason code, and stays at IPL7 until power is completely lost or a node reset is done by the host.

5.3 Port Reset

There are two types of port reset, which are different primarily by the amount of time taken to reset and by the amount of saved state. These are:

- Node reset
- Node halt

5.3.1 Node Reset

Node reset is initiated by writing the *Node Reset (NRST)* bit in the XMI Bus Error Register. This form of reset results in the same sequence of actions as power up, running complete self test for up to 10 seconds.

No state is saved across node resets except for the last MOP Boot message received.

5.3.2 Node Halt

Node halt by the host is initiated by setting the *Node Halt (NHALT)* bit in the XMI Bus Error Register. It is expected that the setting of *Node Halt (NHALT)* must be followed by generating a *Node Reset (NRST)*.

On node halt, the Adapter does the following:

- Set an internal boolean to prevent scheduling of new RTOS tasks.
- Disable FDDI link.
- Stop processing FDDI received frames from the RMC/REM.
- Cancel all scheduled RTOS tasks.
- Set adapter state to *Halted*

CHAPTER 6

ERROR HANDLING

This chapter gives a description of the internal errors detected by the Adapter Manager and reported to the Port Driver.

6.1 Errors

This section describes how internal hardware errors are handled. All detectable fatal errors are non-recoverable and will result in the adapter changing state to Halted and storing error log information into one of four error log entries. Recovery from the Halted state requires the adapter to be reset by the Port Driver in order to continue. There are two catagories of errors:

1. Those errors resulting in the hardware reporting the error in the SPE, AMPE, or WTO bits in the XPST register. For each of these errors, the Host must write XMI Node Reset to bring the Adapter back up. The adapter manager firmware is not running after these errors are detected, thus no state change occurs. These "hardware assist" errors are as follows:
   - AMI-ESP Bus Error. This indicates a fatal error has been detected in the AMI subsystem. The Adapter State is unchanged. The Adapter Manager is vectored to an ISR that loops on self. The device driver will generate the node reset. The ESP asserts the XPST AMPE bit, and then issues an error interrupt to the Host.
   - Watchdog Timer Expiration. The Watchdog Timer was not reset by the Adapter Manager within its' time-out period. This indicates potentially runaway code. The Adapter State is unchanged, the Adapter Manager is vectored to an ISR that loops on self. The ESP asserts XPST WTO and then issues an error interrupt to the Host.
   - SRAM Parity Error. A parity error was detected when accessing SRAM. Adapter state remains unchanged. The 68020 is frozen solid, and no code is executed. The ESP asserts XPST SPE and then issues an error interrupt to the Host.
2. Those errors resulting in the Adapter Manager reporting the error through the XPST register, and putting the adapter in the Halted State. For these errors, the Adapter Manager is vectored to an ISR that performs the following actions:
   1. Notifies firmware of impending halt condition.
   2. Freezes internal packet processing progress (both transmit and receive) through writing CSRs on the PMC and the Parser.
   3. Writes the Port Error Block in the PDB.
   4. Cancels all internal Adapter running tasks.
   5. Logs the error into the one of the four EEPROM Error Log entries.

6. Writes the Halted state code to the Adapter State field of the XPST register. This results in the Host being interrupted by the ESP. To continue the host must reset the adapter.

The following errors cause the Adapter to enter the Halted State. For a complete description of their meaning refer to the XFA Hardware Functional Spec and the ANSI FDDI - SMT Standard.

1. HARDWARE FAILURES (FATAL)
   - CNS DETECTED ERRORS
     - CNS$RMC_PARITY
     - CNS$RMC_PARITY_THRESHOLD (default = 1)
     - CNS$RMC_NP_ERROR
     - CNS$RMC_RECEIVE_STOPPED
     - CNS$RMC_TRANSMIT_STOPPED
     - CNS$MAC_PARITY
     - CNS$MAC_PARITY_THRESHOLD (default = 1)
     - CNS$MAC_NP_ERROR
     - CNS$MAC_RMC_PARITY
     - CNS$MAC_RMC_PARITY_THRESHOLD (default = 1)
   - PARSER DETECTED ERRORS
     - XFA$RMC_PARITY_ERROR
     - XFA$ILLEGAL_CSR_ACCESS
     - XFA$CSR_BUS_PARITY
     - XFA$DATABUS_PARITY
     - XFA$STATE_MACHINE_ERROR
   - ESP DETECTED ERRORS
     - XFA$ESP_PMC_BUS_PARITY
     - XFA$XMI_BYTE_COUNT_ERROR
     - XFA$PBI_BYTE_COUNT_ERROR
     - XFA$TRANSMIT_FORMAT_ERROR
     - XFA$RECEIVE_FORMAT_ERROR
     - XFA$TRANSMIT_OWN_ERROR
     - XFA$RECEIVE_OWN_ERROR
     - XFA$XMI_ERROR
     - XFA$ESP_STATE_MACHINE_ERROR
     - XFA$ESP_IPL_ERROR
   - PMC DETECTED ERRORS
     - XFA$PMC_DETECTED_PARITY

    - XFA$REM_PROTOCOL_ERROR
    - XFA$ESP_WRITE_NOT_OWNED
    - XFA$PMC_STATE_MACHINE_UNKNOWN
  - AMI DETECTED ERRORS
    - RTOS$WATCHDOG_TIMER_EXPIRED
    - XFA$BUS_FAULT_ERROR
    - XFA$CSR_PARITY_ERROR
    - XFA$AMI_ESP_BUS_PARITY
2. SOFTWARE FAILURES (FATAL)
  - CNS$SW_FAULT
  - XFA$SW_FAULT
3. OTHER CONDITIONS (no error log generated)
  - CNS$PC_TRACE_COMPLETED
  - XFA$POWER_FAILURE
  - XFA$ESP_AMI_BUS_ERROR
  - XFA$SRAM_PARITY_ERROR
  - XFA$HALT_RECEIVED (driver issued)

6.2 Errors Detected Prior to the Adapter Entering the Initialized State

Because the Adapter has no knowledge of the Host interrupt vectors prior to the Initialized State, no Host interrupts are issued when errors are detected in the Resetting or Uninitialized States.

The Port Driver will timeout waiting for transition from Resetting to Uninitialized state or from Uninitialized to Initialized state.

CHAPTER 7

PORT MISCELLANEOUS FUNCTIONS

7.1 Maintenance Services - MOP XID/TEST support

The XFA implements maintenance operations as specified in the DNA Maintenance Operations Functional Specification. The description and message formats of these operations can be found in that document.

The XFA supports the following set of maintenance functions for 802 frames.

- Loopback. The XFA decodes Loopback messages and either forwards them to another node or delivers them to the port driver, depending on the content of the message.
- Remote Console. The XFA processes the following types of MOP messages:
  - Request ID. The XFA sends a System ID message to the requesting node in response to a Request ID message.
  - System ID. The XFA sends System ID messages on a regular basis to the Remote Console Server multicast address.
  - Request Counters. The XFA sends a Counters message to the requesting node in response to a Request Counters message.
  - Boot. The XFA causes an XMI bus reset upon receipt of a valid boot message.

With regard to adapter handling of MOP, loopback and 802 Test/XID messages, they are responded to only on the MLA address. The conventions observed are as follows:

- MOP, loopback and 802 Test/XID messages addressed to the appropriate multicast address are responded to only by the appropriate entity on the host. The adapter does not "listen" for any multicast address for these classes of messages.
- MOP and loopback messages addressed to an enabled physical address are responded to from that physical address.
- MOP Request Counters messages return data link counters which are tallied across all enabled users. This involves work on the part of the adapter summing the host-resident and adapter-resident counts into the RDCNTR block.

7.1.1 Mapped Ethernet Frames

The adapter firmware also understands MOP frames in Extended 802.2 format that contain an Mapped Ethernet frame. The first three bytes (as received on the wire) of the PI field contain a code indicating this is a MOP Mapped Ethernet frame. The Ethernet Protocol Type value is contained in the last 2 bytes of the 5 byte PI field. MOP frames received in Mapped Ethernet format are in MOP version 3.0 format, and responses to such requests will also be in that format.

7.1.2 Trigger Boot Mechanism

Trigger booting is initiated by receipt of a MOP Boot message. After the frame has passed the validation checks, the firmware causes an XMI reset, which will cause the host to reboot. Refer to System Console Users Guide regarding this mechanism.

The validation checks are:

- Port is in the *Running* state, and the following checks succeed:
  - Verify frame size is valid
  - Ensure processor is System processor and not Communication processor
  - *Boot Verification Code*, as given in a prior PARAM command, is either zero or matches the code given in the MOP frame.
  - *BOO* flag, as given in a prior PARAM or SYSID command, is set.

7.2 Multiple Physical Address Support

The XFA adapter supports the use of multiple physical addresses. There are two types of addresses:

- "My Long Address". This address is the default address contained in the ROM Default Physical Address. All users by default receive frames with the MLA.
- Addresses other than the MLA address. When requesting access to the FDDI, a user may specify a physical address which is different from the MLA address. The firmware hands the Physical Address specified by the user to the filtering hardware. This set of physical addresses assigned to all defined users is stored in the Parser Physical Address database. A frame with the MLA is no longer delivered to this user.

7.3 Special User Support

7.3.1 The Promiscuous User

The promiscuous user is a special type of user defined by a *USTART* port command which has the *MODE* field set to FC only. The promiscuous user can only enable SMT, LLC, User Implementer and Future Standardization frame delivery. All frames for these enabled FCs received by the adapter will be accepted on behalf of the promiscuous user and forwarded to the port driver. There can only be one promiscuous user. The promiscuous user cannot enable the ifcs bit on startup.

The port expects the port driver to deliver the frames based on the Uindex field of the receive ring entry. If the Uindex field corresponds to the promiscuous user, then the port driver should deliver the frame to the promiscuous user. These frames received for only the promiscuous user have not been filtered by the port. If the Uindex field corresponds to a non-promiscuous user, then the port driver must be aware of the existence of the promiscuous user and should copy the frame to the promiscuous user in addition to the user indicated by the Uindex field.

If the frame received is intended for processing by the adapter manager and a promiscuous user exists, then that received frame is first forwarded to the adapter manager and then a copy is made and queued for transfer to the host receive ring.

7.3.2 The Unknown User

The user index for the Unknown user is 29. An unknown frame is a frame which is addressed to the node (matching physical address) or to a multicast address enabled on behalf of this user but which no other user has specifically requested. This user is defined when the driver wishes to define more than the maximum users allowed by the adapter or when the host software prefers to perform the filtering. All of the users become lumped into the unknown user category and the port driver must filter them. All frames for the unknown user are forwarded to the host. Since filtering is only done on the FC and DA, the port driver must complete the filtering process for delivery to individual users.

Note: The USTART/UCHANGE Parameters are interpreted for the Unknown User; i.e. UPA, Multicast Addresses, Mode and AMC.

7.3.3 The 'AMC' User

The 'AMC' (All MultiCast) user is a special type of user defined by a *USTART* port command which has the *AMC* bit set. If set, frames addressed to any multicast address and the specific LLC filters requested of the AMC user are accepted on behalf of the AMC user.

Finally, if another user has a multicast address enabled and a GSAP which is also enabled by the AMC user, that frame is delivered to this other user with the Multiple Destination bit set. The port driver must then perform further filtering to recognize that the frame is also for the AMC user.

7.3.4 The 'SMT' User

The user index for the SMT user is 30. The SMT user has two uses for the device driver. When the driver wishes to perform a management function, it sends an SMT frame with the Transaction ID field bit 31 set. Any response must have the accompanying bit set upon return. All other SMT packets sent to the host are sent to an SMT application above the driver. Here the driver must multiplex all SMT frames to either the management layer or the SMT application. One final note, any frames addressed to the SMT application or the management layer could have either user index 31 or 30. User index 31 is reserved for the Adapter Manager. Since all SMT frames are first sent to the Adapter Manager, and the Adapter Manager must pass all unsupported SMT frames to the host, the user index may contain the Adapter Managers index. The driver must perform further filtering for SMT frames.

7.4 Frame Filtering

This section describes the frame filtering done by the XFA. The filtering is accomplished according to parameters provided by the host when it issues *USTART*, *UCHANGE* and *USTOP* commands.

Whenever a new multicast address, physical address, protocol type, 802 SAP or PI value is enabled, or whenever a new promiscuous, AMC or unknown user is defined with a *USTART* command, the PARSER DATA BASE is updated to include the new user information. Whenever a user is deleted with a *USTOP* command, the PARSER DATA BASE is modified to remove the user.

The procedure described here is followed for each receive frame.

1. Based on Frame Control (FC) information only: the frame is discarded; forwarded to the Host (if a promiscuous user is defined); forwarded to the Adapter Manager (AM) or subject to further filtering.
2. The frame is next filtered on Destination Address (DA). If there is no DA match, the filtering is continued only if there is an AMC user and the address is a multicast address.
3. Next the frame is filtered based on the FC and DA combination. The frame is discarded, forwarded to either Host or AM, or subject to further filtering based on both the FC and DA fields. Only LLC frames will require further filtering. SMT, MAC, User Implementer and Future Standardization frames are only filtered on FC and DA.
4. The frame is then filtered based on the LLC information. If the frame is SNAP SAP, the frame is filtered on the PID field. Otherwise filtering is performed on the DSAP. If there is no match for either the DSAP or PID, the filtering is continued only if there is an Unknown user existing.
5. The frame is then filtered based on the combination of FC,DA and LLC fields.
6. Next the frame is filtered on information such as UI/XID/Test Field, Command/Response bit, or Class1/User-supplied LLC modes. The frame is discarded, or forwarded to either the Host or AM based on information provided corresponding to this combination of FC,DA, and LLC.
7. Finally, if at any time the frame is expected to be discarded, the PARSER hardware will check for a Promiscuous user requesting this FC. A copy of the frame is sent if the promiscuous user exists.

7.5 Frame Size Checking

The Port Driver must perform minimum frame size checking for all frames received. The minimum sizes of frames are dependent on the particular FC values received.

Table 7–1: Frame Size Checking

FC type	Minimum Size	with Packet Header
SMT	17 bytes	20 bytes
SNAP_SAP	25 bytes	28 bytes
Non SNAP	20 bytes	23 bytes
User Implementer	17 bytes	20 bytes

7.6 Received Stale Frames

There are two cases where the device driver may receive erroneous data on the RECEIVE ring.

7.6.1 Multiple Destination bit Erroneously Set

The Multiple Destination bit (M) may be set erroneously in the RECEIVE ring control field notifying the device driver that this frame is for multiple users. This can happen when the adapter firmware is in the middle of processing a USTART, USTOP or UCHANGE command and some of the parameters overlap parameters with other users. The device driver upon detecting the M bit set must perform complete filtering.

7.6.2 Flushing Frames after USTOP (BOOKMARK)

The second instance where the driver may receive excess frames is when a user is deleted from the filter data base. Frames may be queued for that user index in the adapter's packet memory or on the Host RECEIVE ring before the firmware completely disables the user in the PARSER Database. The device driver must drop all frames addressed to that user after the USTOP has been issued and before receiving the 'bookmark'. Also if the device driver wants to reuse that same user index, then it must be assured that all frames originally destined for the old user of that index are indeed flushed out of the system. This is to prevent the delivery of frames addressed to the old user from going to the new user.

To prevent the driver from delivering these frames to another user which happens to reuse that index, we will use a Receive Ring "Bookmark". When a USTOP is issued, firmware will first update the PARSER database thus removing that user and preventing further frames from entering the adapter. Next, firmware will send the 'Bookmark' which is a unique 3 byte frame addressed to the old user index which will be queued to the end of the host interface queue for the host receive ring. The three bytes of the bookmark frame are all zero. It indicates that all the user frames for that particular Uindex have been flushed from the adapter. The driver must ensure that only one USTOP is outstanding at any one time. Before issuing another USTOP the driver must receive the 'Bookmark'.

When this "bookmark" frame is received by the driver, it can reuse that Uindex.

7.7 Host Interrupts

The adapter generates state transition interrupts, ring transition interrupts and error interrupts. Each of these may be disabled by not supplying interrupt data at initialization time.

Interrupts are generated to the host for the following events:

1. Port state transitions.
2. Transmit or Receive ring entry ownership transition from adapter to port driver for an entry which is further in the ring than the port driver has reached.
3. Command or Unsolicited ring entry ownership transition from adapter to port driver.

There are different interrupt vectors generated by the adapter depending on which of the above events occurs.

The interrupt reason code, provided by the adapter is decoded as specified below:

Table 7–2: XFA Interrupt Vector Codes

Interrupt Cause vs. Interrupt Code	Bit 5	Bit 4
Transmit or Receive Ring	0	0
Command or Unsolicited	0	1
State Transition	1	0
Rsvd	1	1

An interrupt for any ring is considered advice from the adapter. The port driver will actually process the newly owned ring entry when it considers it appropriate.

7.7.1 Receive/Transmit Interrupts

The Host Hibernation Registers described in Chapter 2 are used to determine when interrupts should be generated to indicate receive/transmit work.

After the driver completes servicing the Receive and Transmit rings for all valid entries it will write the Host Hibernation registers (XMIT_HIB_LO and _HI, and RCV_HIB_LO and _HI) to indicate that the host is exiting, i.e. "hibernating". The Hibernation registers indicate the last entry serviced by the host before hibernating. When the adapter places a new packet on the receive ring, its internal pointers no longer equal the drivers hibernation registers. This inequality and knowledge that the host is hibernating is what is used to generate the new interrupt. If a new entry is placed on the receive ring and the host is not hibernating then no interrupt is generated. The belief is that the host is still processing the rings and it is not necessary to interrupt the host again. The same is true for the processing of the transmit ring. It should be pointed out that there is one interrupt mechanism for both the RECEIVE and TRANSMIT rings. The host will process the RECEIVE ring first, followed by the TRANSMIT ring. The adapter only interrupts the host when both of the Hibernation registers are written.

One final note. The host must never completely fill either ring before hibernating. They must ensure at least one empty ring entry. This is necessary since hardware logic generates the interrupt based upon the inequality of the hibernation registers and where it last put its last entry. If the rings are allowed to fill completely then the pointers would be equal and there would be a chance that a transaction could complete without an interrupt being generated.

7.7.2 Command/Unsolicited Interrupts

If the adapter changes ownership for a command or unsolicited ring entry, the adapter logic generates a Command/Unsolicited interrupt. An interrupt is generated for each an every ring entry change.

7.7.3 State Transition Interrupts

If the port state changes, a state transition interrupt is generated. This is the same interrupt vector that is used for error interrupts.

7.8 Port Driver Interrupts

The port driver does not issue interrupts directly to the adapter. However, the act of writing the port registers is visible to firmware. This, essentially, allows the port driver to indicate to the adapter that some type of processing is required.

The presence of new ring entries for adapter service is indicated by the use of Port Ring Control Flag Registers. If new work is queued for the adapter, register writes are performed by the port driver to indicate that the ownership of the entries has changed from host to port.

CHAPTER 8

DIAGNOSTIC OPERATION

8.1 Self Test Procedure

Self test is run on power up or node reset (setting of the *Node Reset (NRST)* bit in the XMI Bus Error Register). It takes less than 10 seconds.

Self test consists of the following parts:

1. Execute self test.
2. Write selftest results to the XPUD Register.
3. Jump to operational firmware where RTOS gets booted and STF gets cleared.

Note: Selftest includes the Built-in-Selftest for the FDDI Chip Set.

8.2 EEPROM Update Utility

There are two mechanisms for updating firmware in the XFA. The first mechanism requires a firmware image to be running in the adapter which passes the diagnostic CRC test, It uses MAINTENANCE commands via the host command ring. The second mechanism is used when the firmware image in the adapter is corrupted. In this instance, kernel update code (which is loaded in manufacturing and is never expected to be updated in the field) is always vectored to upon the detection of image corruption. This kernel code along with host resident code uses XMI based registers as the interface for updates. Due to SRAM size constraints, the host will only update the firmware in multiple sections with no sections greater than 1K bytes. This restriction is independent of the update method used.

8.2.1 Using Maintenance Commands To Update the EEPROM Image

The first method available to the Host for updating firmware is through the EEPROM Update Maintenance Commands. These commands are issued to the adapter through the Command Ring. They allow the Host to provide physically contiguous segments of the EEPROM image within command buffers pointed to by the command ring entries. The Adapter Manager will read these segments from Host memory and copy them to EEPROM. When the final image segment is to be written, the Host will use the EE$WRITELAST command. This command causes the Adapter Manager to copy the image segment into EEPROM and do a CRC check on the EEPROM contents. The resulting CRC is compared with the CRC value provided in the last 4 bytes of the last image segment provided by the Host. If the CRCs match, the command succeeds, and the Host may reset the adapter to cause the new EEPROM image to be loaded into SRAM and be executed. If they do not match, the command fails, and the Host may retry the entire update from the beginning. This is possible since the executing image is running out of SRAM and it is a good image.

8.2.2 Corrupted EEPROM Update Mechanism

If the EEPROM becomes corrupted in some way, the fallback update mechanism is performed by an image of firmware called Kernel Code. Kernel Code is not subject to update through the method described above. It is therefore not subject to corruption through failed updates in the field. For that reason, the update method described below is available to the Host even following a failed update attempt resulting in a corrupted code image being loaded into EEPROM. In that case, if the Kernel CRC test passes, the Kernel code image is still be intact. The adapter must be returned to Manufacturing for Kernel code updates. The corrupted image update method uses the following registers:

```
 31 30                        ...                              0
+---------------------------------------------------------------+
|X|      EEPROM Image Physical Address in Host Memory <39:9>     |  XPD1:
+---------------------------------------------------------------+

 31                          16.15                             0
+-------------------------------+-------------------------------+
|        Image Size (longwords) |       EEPROM Offset  (longword) |  XPD2:
+-------------------------------+-------------------------------+
```

Where:

- <31> NIO.
- <30:0> Image Segment Address <39:9>, the page aligned physical address in host memory of the segment of code to be loaded in EEPROM.
- <15:0> EEPROM Address, the longword offset where the segment is to be loaded in EEPROM on the adapter.
- <31:16> Segment Size, the size of the segment (in longwords) to be loaded in EEPROM.

The kernel update firmware follows the following algorithm:

1. Module resets and 68020 starts running SRAM_CHECK out of EEPROM.
2. If OK, then run SRAM_LOAD. Calculate and compare checksum in parallel.
3. If OK, then proceed to normal start.
4. Otherwise, write Bad_Checksum in the XPUD register.
5. Jump to Kernel code and poll the ESP internal registers to detect when the Host writes the EEPROM_UPDATE register.
6. On EEPROM_UPDATE write, get the base physical address and length of code in the host to be written into EEPROM from XPD1 and XPD2.
7. Read the image segment specified and write it to the specified EEPROM locations.
8. If EEPROM written OK then write EED to XPST, and poll for another EEPROM_UPDATE write.
9. The update image continues loading until all segments have been updated.

   After loading the complete EEPROM Image:

   - The driver should reset the adapter, forcing self test to be run (ie go back to step 1)

8.3 The ESP Special Test

This section describes the test run by the Adapter Manager during the INIT command. The test is designed to extend the coverage on the ESP chip and the XMI Corner that could not be checked by Self-Test. The test takes approximately 1 second to run, most of which is the time required to bring up the link.

8.3.1 Overview

The ESP Special Test is aimed at testing the ESP and the XMI Corner of the DEMFA module by: testing the write/read capability of the adapter to host memory; setting up a packet in host memory and having the ESP transmit the packet through the Active Bulkhead's CDC loopback path and back to host memory; and finally looping another packet that forces the ESP to do receive chaining.

This test is called by the firmware in response to the host issuing an adapter INIT command. This test assumes that the firmware has completely initialized the adapter and has verified the integrity of the Port Data Block (PDB). Before issuing the INIT command, the driver must set up 2 transmit entries (both pointing to same buffer is ok), and 2 receive entries. The buffers must be at least 512 bytes of contiguous physical memory. The driver must also ensure that the 3rd transmit and 3rd receive entries OWN bits are set to indicate host ownership. This is necessary so that the ESP chip does not prefetch past our allocated entries.

The running of the test is completely invisible to the driver, since all host interrupts are turned off during the test. The Driver is not required to do any checking after the test is done. The Driver can assume that the adapter will be pointing back at the top of the host rings when the test is finished. However, the Driver is responsible for re-initializing at least the first three entries of both rings again, since the test does not restore the entries to their original content. The Driver (or host based diagnostic) knows that the test passed by getting the state transition to the INITIALIZED state. The Driver detects an error either by a timeout waiting for the state transition or more likely, the adapter transitioning to the HALTED state. On error, the state qualifier in the XPST register indicates that the ESP Special Test failed. The XPUD register's bit 3 indicates if it was the ESP SPECIAL TEST that failed. If set, the test failed, if clear the test passed. A specific error code is written to the XPUD error field if an error is detected. In addition, the XPST Port State Qualifier field is set to "ESP Special Test Failed". Refer to the error code appendix of this document for the definition of the PSQ error field. Refer to the DEMFA Self Test Functional Specification for the XPUD field definitions.

8.3.2 Test Flow

The following pseudo code describes the flow of the ESP Special Test.

```
BEGIN
ret_code = 0
set up the packet header (set SA=DA=MLA)
set special_test failed bit in the XPUD
mask ESP interrupts at the AMI so this test reports error
disable host interrupts at ESP chip
force fc_da for SMT frames to host via parser database
enable the fddi link (wait for FRA bit in XPST, appx. 600ms)
if (f$pdb_access_test() != pass)
  THEN ret_code = failure code
if (f$page_packet_loopback() != pass)
  THEN
    ret_code = failure code
    disable the link
    set loopback done (LBD) bit in ESP chip
if (f$rcv_chain_loopback() != pass)
  THEN
    ret_code = failure code
    disable the link
    set loopback done (LBD) bit in ESP chip
disable the fddi link (clear the FRA bit in XPST)
force fc_da for smt frames to AMI via parser database
put ret_code in XPUD
if (all tests passed)
  THEN
    unmask ESP ints at AMI
    clear the special_test failed bit in XPUD
  ELSE
    put failing area code in XPUD field
enable host interrupts
return (ret_code)
END
```

8.3.3 PDB Accessability Test

The Adapter Manager instructs the ESP chip to write four longword patterns to the Adapter Reserved area of the Port Data Block. The pattern set will ensure the integrity of the data path from the on board ESP chip to host memory. The Adapter Manager then reads and verifies the test area in the Port Data Block.

8.3.4 Page Packet Loopback Test

This test sets up a packet in the first transmit entry of the host's transmit ring. The packet is slightly less than 512 bytes so that when the looped packet is received, and the MAC tags on a 4 byte CRC at the end, the packet will still fit in a page of memory. The Adapter Manager then instructs the ESP chip to loop the packet, the ESP wakes up, transmits the packet and the packet is looped backed to the host receive ring. The loopback point for the packet is the CDC chips on the Active Bulkhead. The Adapter Manager then verifies that the host transmit and receive descriptors as well as the packet data are correct.

8.3.5 Receive Chained Loopback Test

This test is very similar in setup to the page loopback test, but checks that the ESP chip can chain a packet into two receive buffers. A small packet is set up in the 2nd transmit entry (ESP chip wants to use the 2nd entry because it has already prefetched it), the buffers size of the first receive entry is made smaller than the packet size, and the ESP is then requested to loop another packet. Since the packet was too large to fit into the first receive entry, the

ESP chip chains it to the 2nd receive buffer. The Adapter Manager then verifies that the host transmit and receive descriptors as well as the packet data are correct.

8.3.6 Driver Requirements

The following is a list of requirements for the Driver so it can support the ESP Special Test. The Driver must do these prior to issuing an INIT command to the adapter or an error will be returned.

- Allocate the first 2 transmit descriptors to the test.
- Allocate the first 2 receive descriptors to the test.
- Descriptors must each point to a physically contiguous buffer.
  - The buffers must be at least 512 bytes.
  - Both transmit descriptors can point to the same buffer.
- The 3rd entry of both rings must be host owned.
- Set the LST bit for the 1st segment of each of the 2 receive entries.
- The 2 transmit and 2 receive entries should be adapter owned.

8.3.7 Error Codes

The following error codes are returned in bits 16 to 23 of the XPUD register. The Driver is not required to look at these to detect an error condition, they are just listed here for convenience.

Test #	Error #	Suspect	Error Description
			ESP SPECIAL TEST ERRORS - PDB access
–	51	XMI CORNER, ESP	test 1, PDB W/R test failed
			ESP SPECIAL TEST ERRORS - 1 page packet loopback
–	52	ESP	test 2, LBD bit in ESP not set
–	53	ESP	test 2, receive[RCV] descriptor incorrect or ERS set
–	54	ESP	test 2, RCV packet header incorrect
–	55	ESP	test 2, RCV packet data incorrect
–	56	ESP	test 2, TX own bit error
			ESP SPECIAL TEST ERRORS - RCV chain packet
–	57	ESP	test 3, LBD bit in ESP not set
–	58	ESP	test 3, 2nd RCV descriptor incorrect

Test #	Error #	Suspect	Error Description
–	59	ESP	test 3, RCV descriptor incorrect or ERS set
–	5A	ESP	test 3, RCV packet header incorrect
–	5B	ESP	test 3, RCV packet data incorrect
–	5C	ESP	test 3, TX own bit error
			ESP SPECIAL TEST ERRORS - general errors
–	5D	ESP	HOST gave too small buffer segment for test
–	5E	ESP	error accessing PARSER database
–	5F	ESP	CNS error detected

APPENDIX A

COMMAND, UNSOLICITED AND MAINTENANCE OPCODES

This is a list of valid opcodes given in command buffer command ring entries.

Table A–1: Command Ring Opcode Values

Value	Opcode	Action
0	NOP	No operation
1	SYSID	Define SYSID message contents
2	PARAM	Set/read port parameters
3	STATUS	Check DEMFA Status
4	RDCNTR	Read data link counters
5	SET_SMT	Change FDDI Ring Parameters
7	USTART	Start a new user
8	UCHANGE	Redefine an existing user
9	USTOP	Stop an existing user
10	MAINT	Perform specified maintenance command

This is a list of valid opcodes given in unsolicited ring entries.

Table A–2: Unsolicited Ring Opcode Values

Opcode	Indication
F005	Ring Init Initiated
F006	Ring Init Received
F007	Ring Beacon Initiated
F008	Duplicate Address Test Failure
F009	Duplicate Token Received
F00A	Ring Purger Error
F00B	Bridge Strip Error
F00C	Ring Op Oscillation
F00D	Directed Beacon Received
F00E	PC Trace Initiated
F00F	PC Trace Received
F019	Transmit Underrun
F01A	Transmit Failures
F01B	Receive Overruns
F022	LEM Reject
F023	EBUFF Error
F024	LCT Reject

This is a list of valid maintenance opcodes given in a maintenance command buffer.

Table A–3: Maintenance Command Sub-Opcode Values

Value	Opcode	Action
1	DEBUG	Debug Maintenance Command
2	EE$Clear_log	Clear EEPROM Error Log
3	EE$Read_log	Read EEPROM Error Log
4	EE$Disable_log	Disable EEPROM Error Logging
5	EE$Enable_log	Enable EEPROM Error Logging
6	EE$ReadParam	Read EEPROM parameter data
7	EE$WriteEEPROM	Write EEPROM contents
8	EE$Writlast	Write EEPROM last image segment
9	EE$ReadEEPROM	Read EEPROM contents
10	Read$Boot	Read MOP Boot message

This is a list of valid debug opcodes given in a maintenance command buffer.

Table A–4: Debug Command Sub-Opcode Values

Value	Opcode	Action
1	Read$Status	Read module status message

APPENDIX B

PORT STATE QUALIFIERS AND LED STATES

This is a list of state qualifiers given in the XPST register. These are significant after power up or reset and after fatal port error (shutdown or failure to initialize).

Table B–1: Port State Qualifiers

Qualifier	Valid for States	Meaning
0	All	No error
1	*Uninitialized*	Port initialization failed, adapter failed self test
2	*Uninitialized*	Port initialization failed, invalid Port Data Block address in XPD1 and XPD2 Registers Block
3	*Uninitialized*	Port initialization failed, Port Data Block not valid
4	*Uninitialized*	EEPROM contents are invalid
5	*Uninitialized*	Shutdown attempted when port not in the *Initialized*, *Running*, or *Maintenance* state.
6	*Uninitialized*	Initialization attempted when port not in *Uninitialized* state
7	*Uninitialized*	Invalid command ring
8	*Uninitialized*	Invalid transmit ring
9	*Uninitialized*	Invalid unsolicited ring
10	*Uninitialized*	Invalid receive ring
11	*Uninitialized*	Unrecoverable XMI failure, including memory error
12	*Uninitialized*	ESP special test failed
13	*Halted*	Node Halt issued
14	*Halted*	Fatal firmware internal error
15	*Halted*	Fatal hardware internal error
16	*Halted*	FDDI State has gone to Broken
17	*Halted*	Parser Database error
18	*Halted*	Invalid adapter state
19	*Halted*	PC TRACE PATH test
20	*Halted*	CAM delete error
21	*Halted*	CAM read error
22	*Halted*	CAM write error
23	*Halted*	CAM compare error
24	*Halted*	CAM software reset failed
25	*Halted*	Power fail
26	*Halted*	AMI parity error
27	*Halted*	Watchdog timeout error
28	*Halted*	DPA ROM CRC error
29	*Halted*	Bus Fault

This is a table describing the LED states as defined in the XPST register.

Table B–2: LED State Table

State	LED1(STF)	LED2	Description
0	OFF	OFF	T2027 is most likely failing module
1	OFF	ON	The Active bulkhead or cable is the most likely failing unit
2	ON	BLINKING	ESP Special Test has not been initiated
3	ON	OFF	ESP Special Test failed...T2027 most likely failing unit
4	ON	ON	All Tests pass...module OK

APPENDIX C

ERROR CODES

This is a list of error codes returned in the corresponding entries for transmit frames, receive frames, command buffers and unsolicited frames.

C.1 Transmit Error Codes

These are the values in the transmit ring entry error code, returned for a transmit frame.

Table C–1: Transmit Errors - Error Codes in the Transmit Ring

ERC Value	Meaning
0	No Error
20(Hex)	Format Error

C.2 Receive Error Codes

These are the values in the receive ring entry error code, returned for a receive frame.

Table C–2: Receive Errors - Error Codes in the Receive Ring

ERC Value	Meaning
0	No Error
1	Format Error - Initial entry lacks SOP
4(Hex)	Format Error - Byte count less than one page, no EOP for the entry.
8(Hex)	Format Error - Byte count not exhausted, and EOP found; or for an additional segment, byte count not exhausted and SOP found.

C.3 Command Error Codes

These are the values in the command ring entry error code, returned for a port command.

Table C–3: Port Command Errors - Error Codes in the Command Ring

Value	Meaning
0	No error
1	Buffer length error
2	Wrong Adapter State
3	Buffer transfer failed
4	Invalid command
5	Invalid opcode
6	Invalid loopback command
7	Valid command failed

C.4 Unsolicited Error Codes

Table C–4: Unsolicited Errors - Error Codes in the Unsolicited Ring

ERC Value	Meaning
0	No Error
1	Format Error

C.5 Maintenance Command Error Codes

Table C–5: Maintenance Command Errors - Error Codes in the Command Buffer

ERC Value	Meaning
0	No Error
1	CRC failure
2	No Boot message received
3	EEPROM write size error
4	Unable to store Boot message

APPENDIX D

XFA MINIMA / MAXIMA

This is a list of XFA minimum and/or maximum values for various port parameters.

D.1 Port Parameters

Table D–1: XFA Maximum values

Parameter	Maximum Value
Maximum Users	32
Maximum LLC configurations (SAPs and PI)	64
Promiscuous mode users	1
Destination addresses (Individual and Multicast)	64
Maximum Addresses per User Index	16
Maximum Protocol Id's per User Index	1
Maximum GSAP's per User Index	7
Maximum Individual SAP's (ISAP's) per User Index	1

D.2 Host Ring Sizes

This table gives the buffer size for each type of unsolicited indication.

Table D–2: Host Ring Sizes

Opcode	Maximum Size	Entry Size
Transmit Ring	4088 Bytes	8 Bytes
Receive Ring	4080 Bytes	16 Bytes
	4064 Bytes	32 Bytes
	4080 Bytes	48 Bytes
	4032 Bytes	64 Bytes
Command Ring	4096 Bytes	8 Bytes
Unsolicited Ring	4096 Bytes	16 Bytes

D.3 Command Buffer Sizes

This table gives the buffer size for each type of command.

Table D–3: Command Buffer Sizes

Opcode	Minimum Size	Maximum Size
NOP	4	4
SYSID	4	516
PARAM	376	376
STATUS	108	108
RDCNTR	324	324
USTART	168	168
UCHANGE	168	168
USTOP	4	4
SET SMT	24	24

D.4 Transmit Frame Sizes

Table D–4: Transmit Frame Sizes

Characteristics	Minimum Length	Maximum Length
Non-LLC Frame, CRC generated by port driver	17	4495
SAP Frame, CRC generated by port driver	20	4495
SNAP SAP Frame, CRC generated by port driver	25	4495

D.5 Receive Buffer Sizes

Receive buffers associated with a single receive ring entry must be long enough to contain the maximum size, host-specified FDDI frame. This length is specified in the Port Data Block parameter *Max Receive Size*.

D.6 Maintenance Command Buffer Sizes

This table gives the buffer size for each type of MAINTENANCE command.

Table D–5: MAINTENANCE Command Buffer Sizes

Opcode	size
READ STATUS	184
CLEAR LOG	8
READ LOG	520
DISABLE LOG	8
ENABLE LOG	8
READ PARAM	24
WRITE EEPROM	20
WRITE LAST	20
READ EEPROM	20
READ BOOT	420

APPENDIX E

XFA REGISTER ADDRESSES

Table E–1: XFA Register Addresses (seen from the XMI)

Address	Mnemonic	Description
bb + 0	XDEV	XMI Device Register
bb + 4	XBER	XMI Bus Error Register
bb + 8	XFADR	XMI Failing Address Register
bb + 2C	XFAER	XMI Failing Address Extension Register
bb + 100	XPD1	Port Data Register 1
bb + 104	XPD2	Port Data Register 2
bb + 108	XPST	Port Status Register
bb + 10C	XPUD	Power Up Diagnostic Register
bb + 110	XPCI	Port Control Initialize Register
bb + 114	XPCS	Port Control Shutdown Register
bb + 118	XMIT_FL	Transmit Control Register
bb + 11C	RCV_FL	Receive Control Register
bb + 120	CMD_FL	Command Control Register
bb + 124	UNSOL_FL	Command Control Register
bb + 128	XMIT_HIB_LO	Transmit Host Hibernation Register <31:00>
bb + 12C	XMIT_HIB_HI	Transmit Host Hibernation Register <39:32>
bb + 130	RCV_HIB_LO	Receive Host Hibernation Register <31:00>
bb + 134	RCV_HIB_HI	Receive Host Hibernation Register <39:32>
bb + 138	EEProm_Update	EEprom Update Register

Where "bb" refers to the base address of a node. The base address of a given node on the XMI is computed as (physical address in I/O space):

2180 0000 (hex) + 80000 (hex) * NodeID

APPENDIX F

PORT DRIVER/FDDI PACKET HEADER

This is the default packet header used by the adapter for transmitted packets.

F.1 FDDI Packet Header BYTE 0

Table F–1: BYTE 0

Bits	Binary Value	Meaning
7:6	00	RSVD
5:4	10	Token is unrestricted
3	0	Mode is asynchronous
2	0	Ring must be operational before transmit
1	0	Send frames in normal order
0	0	RSVD

F.2 FDDI Packet Header BYTE 1

Table F–2: BYTE 1

Bits	Binary Value	Meaning
7	0	RSVD
6	0	Release token in normal mode
5	1	Append CRC
4:3	11	Release same kind of token received
2:0	000	Do not send extra FCS

F.3 FDDI Packet Header BYTE 2

Table F–3: BYTE 2

Bits	Hex Value
All	0