EtherTalk™ Interface Card Preliminary Note

Final Draft

5/29/87

Apple User Education

APPLE COMPUTER, INC.

20525 Mariani Ave.
Cupertino, California 95014
(408) 996-1010

The information in this document reflects the current state of the product. Every attempt has been made to verify the accuracy of this information, however, it is subject to change. Preliminary notes are released in this form to provide the development community with essential information in order to facilitate work on third-party products.

Table of Contents

List of Figures

List of Tables

Introduction

This document is intended for application developers writing communication software for Local Area Networks (LANs) using the Institute of Electrical and Electronics Engineers (IEEE) 802.3 standards. The document contains programming information for the EtherTalk Interface Card™. It is assumed that programmers developing applications for networking environments are familiar with the software environment they will use. Some familiarity with LANs is also assumed.

Product Overview

The EtherTalk Interface Card is a network adapter card that is used on the Macintosh™ II computer to allow connection to Ethernet® or Thin Net. Ethernet attachment is through a standard drop cable from a transceiver attached to an Ethernet backbone network cable. Thin Net, which uses RG-58A/U coaxial cable, provides a low cost network that is easy to install and that can be attached to standard Ethernet cabling through a repeater. The EtherTalk card has an onboard transceiver for use with Thin Net.

Three Very Large System Integration (VLSI) circuits from National Semiconductor Corporation perform the main functions of the card. A Network Interface Controller (NIC) circuit implements all Media Access Control (MAC) layer functions for transmission and reception of packets in accordance with the IEEE 802.3 standard. A Serial Network Interface (SNI) circuit provides the encoding and decoding functions for the Ethernet or Thin Net Local Area Networks. A Coaxial Transceiver Interface (CTI) drives the Thin Net LAN. The card is programmed by accessing registers and memory on the card from the bus interface on the Macintosh II system. Information is written to and read from the card by using dual-ported RAM located on the EtherTalk card.

> *Note:* Hardware on the EtherTalk card is subject to change. Software that writes directly to the hardware may be affected by hardware changes and may require software modifications. It is recommended that Apple standard drivers be used where possible. Apple Computer will be providing such drivers for the Macintosh II operating system and for A/UX™, Apple's implementation of the UNIX® operating system.

Where to Find Additional Information

Additional information can be found in the following publications:

- *ANSI/IEEE 802.3 Carrier Sense Multiple Access with Collision Detection*
- National Semiconductor Corporation publication *DP8390/NS32490 Network Interface Controller*
- National Semiconductor Corporation publications *Advanced Peripherals IEEE 802.3 Local Area Network Guide* and *Datasheet Addendum* (March, 1987)
- *Ethernet Blue Book*
- Apple Computer publication *EtherTalk Card Installation Guide*
- Apple Computer design specification proposal *Macintosh OS Ethernet Driver*
- Apple Computer publication *A/UX Network Applications Programming*
- Apple Computer publication *A/UX Programmer's Reference*
- Apple Computer publication *Building A/UX Device Drivers*
- Apple Computer publication *Macintosh II and Macintosh SE Cards and Drivers*

The last four publications on this list are available from the Apple Programmers and Developers Association (APDA).

Network Architecture, Protocols, and Interfaces

The development of small but powerful computer systems by different manufacturers has caused a unique problem. Because each manufacturer had a different operating system, communications between different computer systems was difficult. In addition, numerous private and public data networks were developed, and the protocols developed by the various networks were incompatible. Communications between the systems and networks was possible only if a common set of rules was developed. Systems and networks require communications standards that provide the following:

- standard protocols and interfaces with common mechanisms for communications among systems
- a standard approach, called a *network architecture*, that defines the interactions and relationships between network services and functions by common interfaces and protocols

The International Standards Organization (ISO) recommended a standard in the form of a seven-layer model for network architectures, known as the Open Systems Interconnection (OSI) model. This model is an example of a network architecture consisting of a hierarchy of modules, or layers, each of which performs defined functions. Figure 1 shows the seven layers of the OSI model.

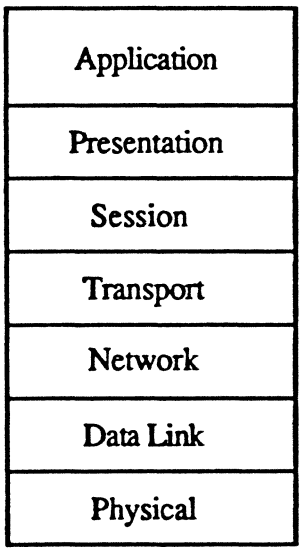


Figure 1. OSI Layers

In a layered network architecture such as the OSI model, the function of each module and the relationships between modules are defined. This approach provides a common set of rules that define the way network *nodes* (network users' computers) must interact in order to communicate and share information.

A layered architecture defines two kinds of relationships between modules, *interfaces* and *protocols*. Interfaces define relationships between modules operating within a network node. A module in one layer will interface with a module below it to receive a *service*. For example, in the OSI model, the physical layer, which defines the physical connection for

data transmission, interfaces with the data link layer, which establishes a communications path between network nodes over the physical channel. The data link layer, in turn, interfaces with the network layer, which addresses messages and routes them across intervening nodes to their destination. Each layer provides a certain subset of services to the overall network functions. Protocols define relationships between equivalent modules; they also define message formats, the rules for information exchange.

The OSI model is often used as a reference point for the architecture of LANs such as Ethernet and Apple Computer's AppleTalk.

EtherTalk Card Media Access Method

The media access method used by the EtherTalk Interface Card is Carrier Sense Multiple Access with Collision Detection (CSMA/CD). The CSMA/CD access method is a means by which two or more nodes share a common bus transmission medium. In this implementation, several Macintosh II computers equipped with the EtherTalk card are attached to the same Ethernet cable, and CSMA/CD enables each to detect whether the cable is in use before attempting to use the cable for communication.

The multiple access feature of CSMA/CD lets any node send a message immediately upon sensing that the network is free of traffic. The carrier sense feature, also called *listen before talking*, is the ability of each node to detect any traffic on the network. Nodes will not transmit whenever they sense that there is traffic on the network. However, given the time it takes a signal to travel across the network (called propagation delay), two nodes could detect that the network is clear simultaneously, or at nearly the same time, because each will not have detected the signal of the other. This situation is called a *collision* between the two messages. Collision detection is the ability of a transmitting node to sense a change in the energy level on the network and to interpret the change as a collision. Upon detecting a collision, each node *backs off* and abandons its transmission. After waiting for a period of time the node tries to transmit again. If there are successive collisions, the nodes back off and wait a longer period each time. The nodes also send a short burst of noise, called a *jam*, to ensure that all other nodes involved have detected the collision.

Media Accessed by the EtherTalk Card

The EtherTalk Interface Card network connection supports the following two media types defined by the IEEE 802.3 standard:

- Ethernet
- Thin Net

Ethernet (the IEEE 10BASE5 medium) connects to the card with a transceiver drop cable using the DB15 connector on the card. Thin Net (the IEEE 10BASE2 medium) connects to the card using a BNC connector on the card. Media type selection is determined by a jumper on the EtherTalk card. Both media are accessed by means of CSMA/CD signalling on the EtherTalk card.

For more information regarding installation, consult the *EtherTalk Card Installation Guide* available from Apple Computer.

Operating Systems and Drivers

The EtherTalk Interface Card runs on a Macintosh II under the A/UX operating system. A detailed discussion of the A/UX operating system or device drivers is beyond the scope of this document. Please refer to the Apple publications *Building A/UX Device Drivers*, *A/UX Programmer's Reference*, and *A/UX Networking Applications Programming*, all available from the Apple Programmers and Developers Association (APDA).

A driver that will be with used the Macintosh II operating system is currently being developed by Apple Computer. The interface will be similar to the AppleTalk Link Access Protocol (ALAP) part of the AppleTalk driver.

Information on this driver can be found in the Apple publication *Inside Macintosh, Volume 2*, Chapter 10, "The AppleTalk Manager" and in the Apple design specification proposal *Macintosh OS Ethernet Driver*. This driver will be documented in a future preliminary note. Interested persons should contact Apple Computer Developer Technical Support.

EtherTalk Card Hardware Description

The EtherTalk Interface Card is a non-intelligent Ethernet adapter for the Macintosh II computer. The card uses a LAN chipset from National Semiconductor Corporation. The three LAN chips are the Network Interface Controller (NIC), Serial Network Interface (SNI), and a Coaxial Transceiver Interface (CTI). The card has 16K of dual-ported Random Access Memory (RAM) and 32K of Read Only Memory (ROM). The local memory allows back-to-back packet reception with multipacket buffering. The architecture of the card is shown in Figure 2.

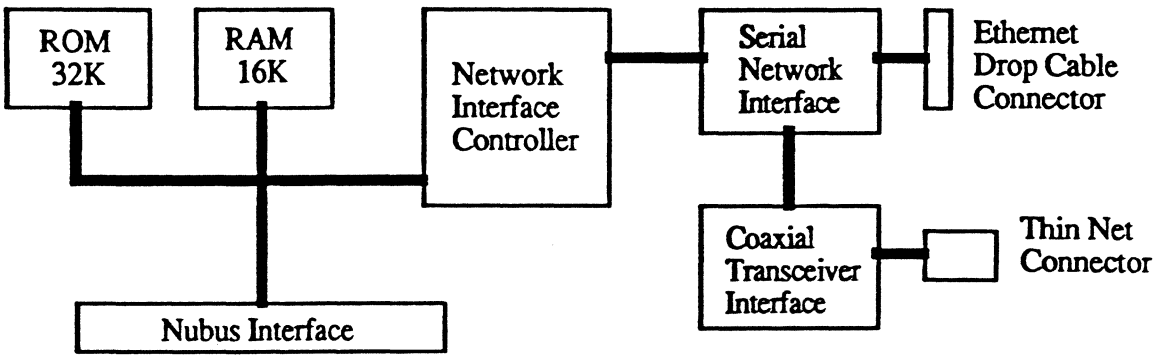


Figure 2. EtherTalk Card Architecture

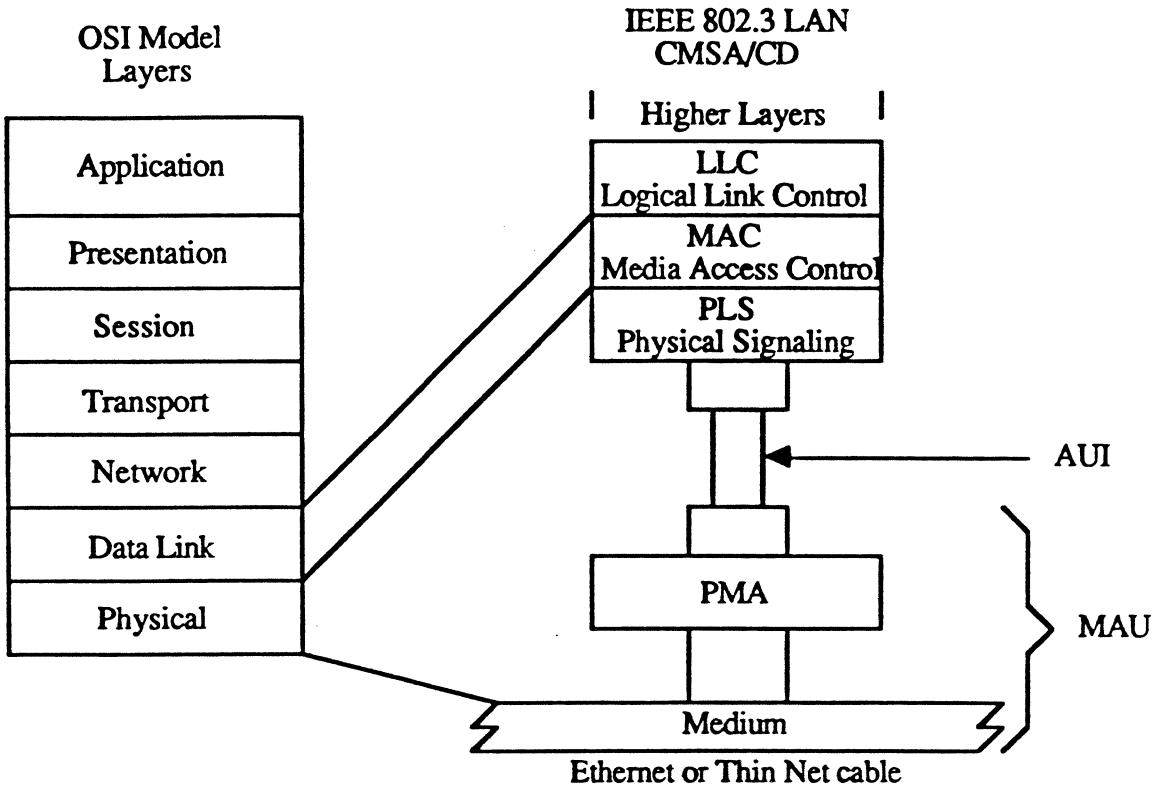
The EtherTalk card uses the Apple's implementation of the Nubus™ interface. More information on the Apple's implementation of Nubus can be found in the Apple Computer publication *Macintosh II and Macintosh SE Cards and Drivers* available from APDA.

The CTI chip is used as a coaxial line driver/receiver for Thin Net LANs. The CTI is not used when attaching to an Ethernet backbone cable by means of an external transceiver. The selection is made with a jumper on the EtherTalk card.

The SNI chip provides the encoding and decoding functions for Ethernet or Thin Net LANs. The SNI also provides a collision signal translator and a diagnostic loopback circuit.

The NIC chip is the heart of the LAN chipset. The NIC performs all Media Access Control (MAC) layer functions for transmission and reception of Ethernet packets. The NIC provides buffer management that supervises storage of received packets in the local memory. During packet transmission, the NIC generates and appends the preamble and sync byte to the transmitted packet. Also, the NIC will optionally compute and append Cyclic Redundancy Check (CRC) bytes. During reception, the NIC decodes and filters addresses and performs CRCs. More programming information about the NIC can be found in the National Semiconductor specification document titled *DP8390/NS32490 Network Interface Controller.*

The relationship of the hardware and services provided by the LAN chipset is shown in Figure 3.



AUI - Attachment Unit Interface (Ethernet drop cable)
MAU - Medium Attachment Unit
(Ethernet Transceiver and drop cable
or CTI and Thin Net cable)
PMA - Physical Medium Attachment
(transceiver or BNC connector)

Figure 3. Hardware Model

Local Memory

The local memory consists of 16K of static RAM segmented into *transmit* and *receive buffers* by setting registers in the NIC. The segments are further divided into 256-byte *pages*. Some number of pages (under driver control) is used for a transmit buffer. The remaining pages are used for a receive ring buffer. Page Start and Page Stop registers establish a buffer that forms a continuous address space. As the last address is reached (set up by the Page Stop register), the next memory location wraps around to the start of the buffer (set up by the Page Start register) to form the receive ring buffer.

The local ROM memory is 32K in size and contains the Ethernet address and Nubus card slot information. Direct access of the ROM is not usually necessary because these services are provided by the slot library available in A/UX.

Address Assignments

Figure 4 shows the address map of devices on the EtherTalk card.

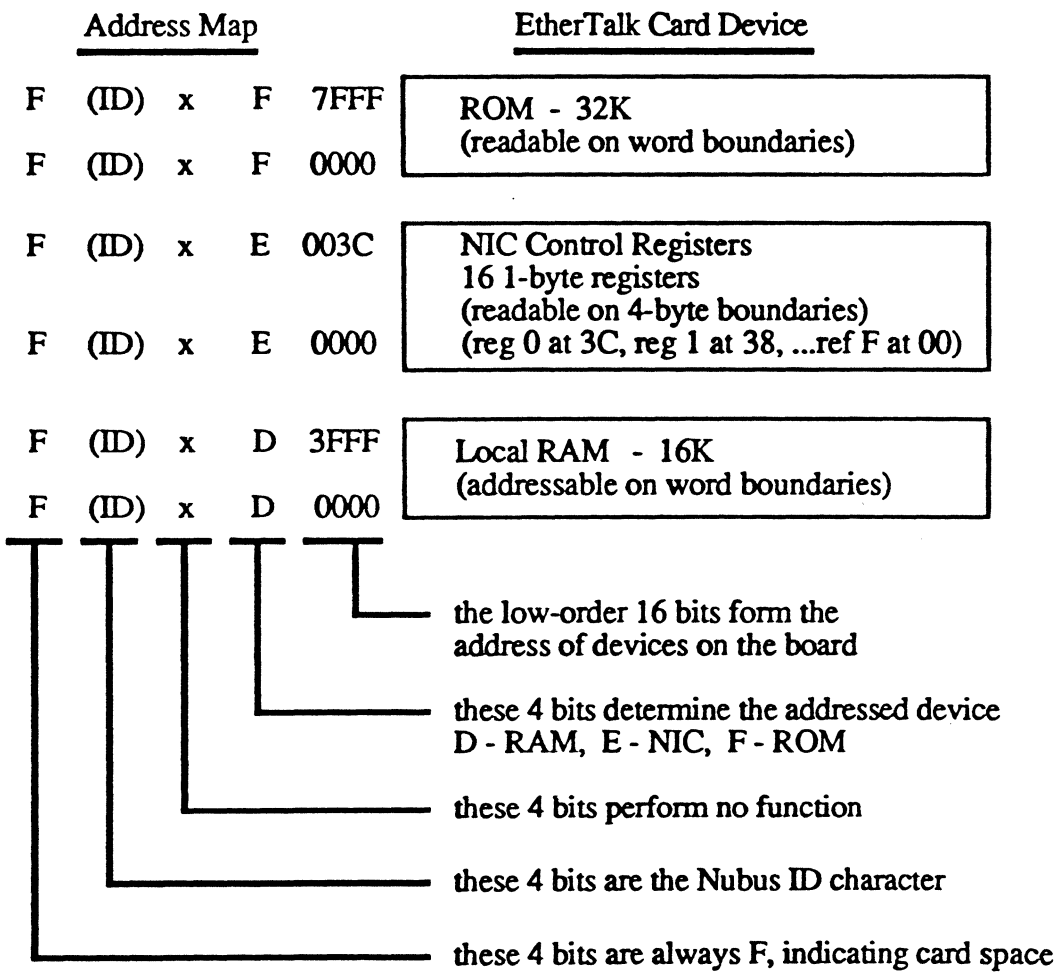


Figure 4. Address Assignments

NIC Register Addresses

NIC registers are divided into three pages. The content of the highest-order bits in the Command register (PS0 and PS1) defines which page of registers is being read from and written to for addresses E0000 through E003C. Page 0 registers are those registers commonly accessed during normal operation of the NIC. Page 1 registers are accessed during the initialization process. Page 2 registers should only be accessed for diagnostic purposes and should not be modified during normal operation. For complete definition of the register terms in the tables, consult the National Semiconductor Corporation publication *DP8390/NS32490 Network Interface Controller.*

The following tables show the registers used for programming the NIC. Table 1 displays the NIC Page 0 register addresses on the EtherTalk card.

Table 1. Page 0 Address Assignments (PS1=0, PS0=0)

Address	Page 0 Read	Page 0 Write
E003C	Command Register	Command Register
E0038	Current Local DMA Address (CLDA) 0	Page Start Register
E0034	Current Local DMA Address (CLDA) 1	Page Stop Register
E0030	Boundary Register	Boundary Register
E002C	Transmit Status Register	Transmit Page Start Register
E0028	Number of Collisions Register	Transmit Byte Count Register (TBCR) 0
E0024	First In First Out (FIFO) Register	Transmit Byte Count Register (TBCR) 1
E0020	Interrupt Status Register	Interrupt Status Register
E001C	Current Remote Data Address (CRDA) 0	Remote Start Address Register (RSAR) 0
E0018	Current Remote Data Address (CRDA) 1	Remote Start Address Register (RSAR) 1
E0014	Reserved	Remote Byte Count Register (RBCR) 0
E0010	Reserved	Remote Byte Count Register (RCBR) 1
E000C	Receive Status Register	Receive Configuration Register
E0008	Counter (CNTR) 0	Transmit Configuration Register
E0004	Counter (CNTR) 1	Data Configuration Register
E0000	Counter (CNTR) 2	Interrupt Mask Register

The byte counts in Transmit Byte Count registers 0 and 1 are combined to create a single count. The byte counts in Remote Byte Count registers 0 and 1 are also combined. Counter 0 is used for frame alignment errors, counter 1 for CRC errors, and counter 2 for missed packet errors.

Table 2 displays the NIC Page 1 register addresses on the EtherTalk card.

Table 2. Page 1 Address Assignments (PS1=0, PS0=1)

Address	Page 0 Read	Page 0 Write
E003C	Command Register	Command Register
E0038	Physical Address Register (PAR) 0	Physical Address Register (PAR) 0
E0034	Physical Address Register (PAR) 1	Physical Address Register (PAR) 1
E0030	Physical Address Register (PAR) 2	Physical Address Register (PAR) 2
E002C	Physical Address Register (PAR) 3	Physical Address Register (PAR) 3
E0028	Physical Address Register (PAR) 4	Physical Address Register (PAR) 4
E0024	Physical Address Register (PAR) 5	Physical Address Register (PAR) 5
E0020	Current Point (CURR)	Current Point (CURR)
E001C	Multicast Address Register (MAR) 0	Multicast Address Register (MAR) 0
E0018	Multicast Address Register (MAR) 1	Multicast Address Register (MAR) 1
E0014	Multicast Address Register (MAR) 2	Multicast Address Register (MAR) 2
E0010	Multicast Address Register (MAR) 3	Multicast Address Register (MAR) 3
E000C	Multicast Address Register (MAR) 4	Multicast Address Register (MAR) 4
E0008	Multicast Address Register (MAR) 5	Multicast Address Register (MAR) 5
E0004	Multicast Address Register (MAR) 6	Multicast Address Register (MAR) 6
E0000	Multicast Address Register (MAR) 7	Multicast Address Register (MAR) 7

The operating system reads the EtherTalk card ROM and installs 6 bytes into the Physical Address registers 0 through 5. The Current Point is a page where a packet is currently being received; it is used to detect packet reception. The Multicast Address registers are initialized to 0XFF because A/UX does not support multicasting.

Programming Guidelines

1. The EtherTalk card uses the Apple Nubus interface. Direct access to local memory is provided to the Macintosh II, so the NIC Direct Memory Access (DMA) channel or the Send Packet command (for automatic uploading of the receive ring buffer) is not used.

2. The Interrupt Status register is reset by writing FFh to itself. This clears the seven low-order bits.

3. Use the Boundary pointer to remove packets from the receive ring and to free buffer space for reception of additional packets. Exercise care to make certain that the Boundary register lags at least one behind the last freed buffer page.

4. Interrupts to the Macintosh II are masked off or enabled via the Interrupt Mask register. The cause of the interrupt is presented in the Interrupt Status register, whether the Macintosh II is masked or enabled. This allows event servicing either through an interrupt service routine, which checks the Interrupt Status register to determine the interrupt cause, or through polling of the Interrupt Status register.

5. Care should be exercised in modifying the NIC Command register. In particular, only a single command function can be modified on a single write to the Command register. For example, to change from Stop mode (21h in the CR) to Start mode with Transmit packet (26h in the CR), first write 22h (changes from Stop mode to Start mode) and then write 26h (adds the Transmit packet command) to the Command register.

Initialization Procedure

The NIC must be initialized before packets can be transmitted to or received from the network. This section presents an example of the steps necessary to initialize the NIC on the EtherTalk card. The values given for registers in the procedure may vary according to the individual application. Initialization values for some registers are not given because the values are application dependent Programming information about the NIC (available from National Semiconductor) should be read thoroughly before attempting to program the NIC registers.

1. Set the Command register to 21h to stop the NIC and select Page 0.

The Command register is used to initiate transmissions and to select register pages.

2. Initialize the Data Configuration register.

The Data Configuration register is used to program the NIC for 8-bit or 16-bit memory interface and establish First In First Out (FIFO) thresholds for transfers between the NIC and card RAM.

3. Clear the Remote Byte Count registers (RBCR0, RBCR1). Remote DNA is not used on the Macintosh II.

4. Initialize the Receive Configuration register.

The Receive Configuration register determines operation of the NIC during reception of a packet as well as determining what type of packets to accept.

5. Place the NIC in Loopback Mode 1 (internal loopback) or Loopback Mode 2 (external loopback).

Writing 02h to the Transmit Configuration register selects internal Loopback mode, 04h will select external Loopback mode.

6. Initialize the Receive Ring buffer. Set the values for the Boundary pointer (BNDRY), the Page Start (PSTART), and Page Stop (PSTOP).

7. Clear the Interrupt Status register by writing FFh to itself.

8. Initialize the Interrupt Mask register for desired interrupts.

The Interrupt Mask register mask interrupts. Each interrupt mask bit corresponds to a bit in the Interrupt Status register. If an Interrupt Mask register bit is set, an interrupt will occur whenever the corresponding bit in the Interrupt Status register is set.

9. Program the Command register for page 1 registers. Initialize the Physical Address registers (PAR0-PAR5), Multicast Address registers (MAR0-MAR5), and the Current Page register (CURR).

These registers can be accessed only when the Command register is set to page 1. The Physical Address registers are used to compare the destination address of incoming packets so that the packets can be accepted or rejected. The Multicast Address registers provide filtering of multicast addresses to determine if the packets are accepted or rejected. The Current Page register contains the first address of the first buffer to be used for reception and is used to restore memory pointers in case of receive errors.

10. Write 22h to the Command register.

This will place the NIC in the Start mode.

11 Initialize the Transmit Configuration register to the intended value.

The NIC is now ready for transmission or reception of packets from the network.

Receive Buffer Ring Overflow Procedure

During heavy network conditions which overflow the Receive Buffer ring, the NIC may not buffer any more packets from the network. To guarantee this will not happen, a software reset must be issued during all Receive Buffer ring overflows. The overflow condition is indicated by the OVW bit in the Interrupt Status register. The bit is set when Receive Buffer ring storage resources have reached the limit set by the Boundary pointer. This procedure will reset the NIC and clear appropriate registers and buffers.

1. Write 21h to the NIC Command register.

This will place the NIC in Stop mode. The NIC will enter the Stop mode only after processing of the current packet is finished. The NIC indicates Stop mode by setting the RST bit in the Interrupt Status register.

2. Remove at least one packet from the Receive Buffer ring.

3. Clear the Remote Byte Count registers (RBCR0, RBCR1).

The NIC requires these registers to be cleared before setting the RST bit.

4. Read the Interrupt Status register to see if the RST bit is set.

The NIC is in Stop mode when the RST bit is set.

5. Place the NIC in Loopback Mode 1 (internal loopback) or Loopback Mode 2 (external loopback).

Writing 02h to the Transmit Configuration register selects internal Loopback mode, 04h will select external Loopback mode. This step is required to properly enable the NIC onto

6. Write 22h to the Command register.

This will place the NIC in the Start mode.

7. The NIC can be brought out of Loopback mode by programming the Transmit Configuration register to the value determined in Step 11 of the Initialization Procedure.

Packet Transmission

The local DMA is used during packet transmission. During the transmission of a packet, the following three registers control DMA transfer:

- Transmit Page Start Address register
- Transmit Byte Count register 0
- Transmit Byte Count register 1

When these registers point to a packet, the NIC receives a command to transmit the packet. During packet transmission, buffer memory data is moved into the FIFO as required. The NIC generates and appends the preamble, synch, and CRC fields of the packet.

The NIC must have a contiguous assembled packet with the following format:

- destination address (6 btyes)
- source address (6 bytes)
- type length (2 bytes)
- data (up to 46 bytes)
- pad (if data is less than 46 bytes)

The transmit byte count includes the destination address, source address, length field, and data. The preamble and the CRC are not included in this count. When data of less than 46 bytes is being transmitted, the packet must be padded to a minimum size of 64 bytes.

Before the packet is transmitted, the Transmit Page Start Address register an the Transmit Byte Count registers 0 and 1 must be initialized. The following steps are necessary to initiate packet transmission:

1. The TXP bit in the Command register is set.

2. The Transmit Status register is cleared.

3. The NIC begins to prefetch transmit data from memory.

Step 3 will not occur if the NIC is currently receiving.