Table of Contents

Section 1

Applications Support 1-1

Section 2

Introduction 2-1

Section 3

DP8390 Evaluation Board 3-2

Cable Installation 3-11

Section 4 - Software Reference

Software Update 4-4

Computer Conferencing 4-5

Demostration Network Software 4-7

NIA Access Software 4-9

Network Load Simulator 4-17

Network Evaluation Software 4-21

Writing Drivers for DP8390 4-25

Section 5 - Hardware Reference

DP8390 Introductory Guide 5-1

Hardware Design Guide 5-9

Guide to Loopback 5-17

StarLAN with the DP8390 5-26

Section 6

Tranceiver Evaluation Kit 6-1

SECTION 1

APPLICATION CONTACTS

National
Semiconductor
Corporation

Objectives
Preliminary
July 1986

OBJECTIVE OF DP839EB CHEAPER/ETHER DEMONSTRATION KIT

The Cheaper/Ether demonstration kit is intended to provide designers with tools for evaluations and development of networking products using the DP839X chip set. The kit supports Ethernet, Cheapernet and Starlan networks as described in the IEEE 802.3 standard. All required documentation has been provided inside this binder, including the circuit diagram, PAL equations and option settings. Software tools are also provided as guides to developing drivers for the DP8390 Network Interface Controller. **It is important to read all hardware and software manuals prior to installing the demonstration kit.**

IN CASE OF PROBLEMS...

If you encounter problems not addressed in the documentation, contact your local National Semiconductor Field Engineer or Field Sales Office. They will provide you with any additional support you may require.

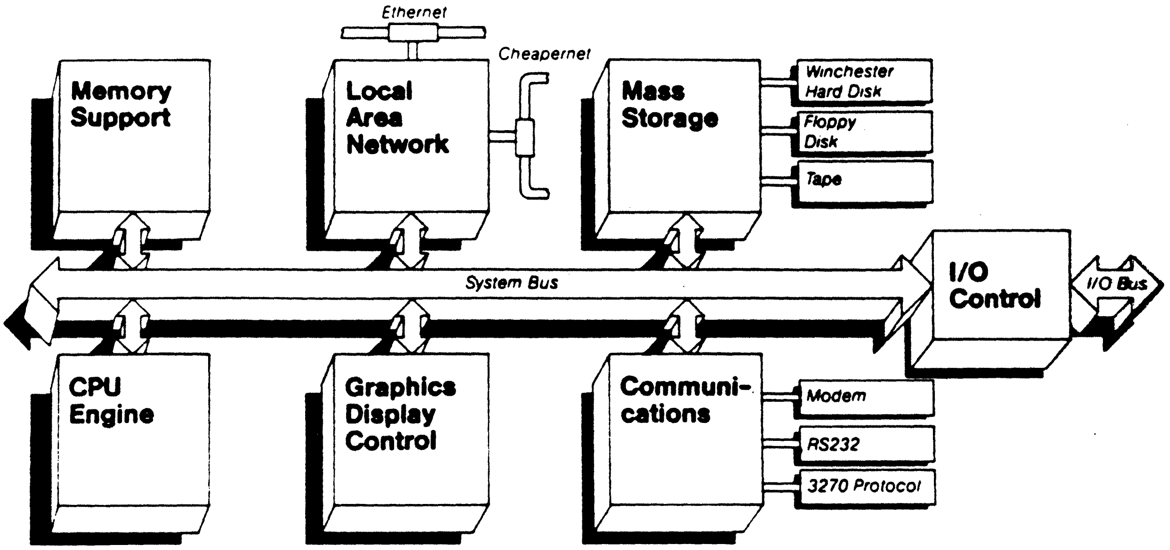
If you have further technical inquiries regarding operation of the DP839X chip set contact National Semiconductor at (408) 721 4247 (for the DP8390) or (408) 721 3857 (DP8392 and DP8391).

SECTION 2

INTRODUCTION

Introduction

Advanced Peripherals



Today's computer systems and information processing needs have created a huge demand for Local Area Networks (LANs). The IEEE 802.3 Standard for Ethernet/Cheapernet and Starlan has become the most popular networking solution. National Semiconductor provides a complete three chip solution for an entire 802.3 design as well as supplying a broad range of products to fill the needs of network design engineers. The chipset consists of the DP8390 Network Interface Controller (NIC), the DP8391 Serial Network Interface (SNI) and the DP8392 Coaxial Transceiver Interface (CTI).

To place your name on the mailing list for design information updates and to receive further information on National's Advanced Peripherals please return the enclosed card or contact your local National Semiconductor Sales Office.

THE CHIPSET FEATURES INCLUDE

DP8390

- Interfaces with 8-, 16-, and 32-bit microprocessor systems
- Implements simple, versatile buffer management
- Utilizes low power microCMOS process
- Includes
  - –Two 16-bit DMA channels
  - –16-byte internal FIFO with programmable threshold
  - –Network statistics storage

DP8391

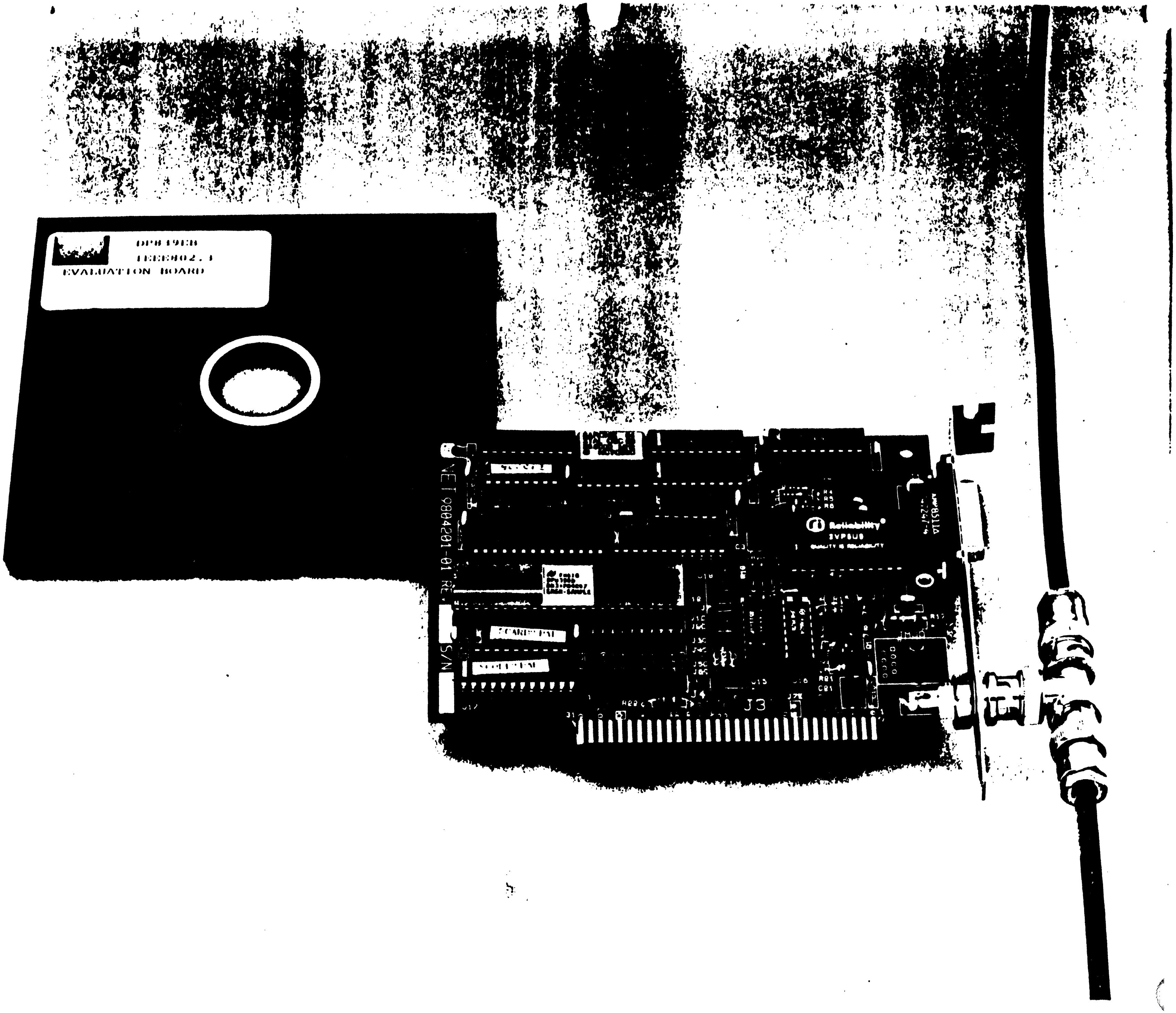
- 10 Mbit/sec Manchester encoding/decoding with receive clock recovery
- Patented digital phase locked loop decoder requires no precision external components
- Decodes Manchester data with up to ± 20 ns of jitter
- Squelch circuits at the receive and collision inputs reject noise
- Connects directly to the AUI cable

DP8392

- Integrates all transceiver electronics except signal and power isolation
- Innovative design minimizes external component count
- Externally selectable CD heartbeat allows operation with IEEE 802.3 compatible repeaters
- Designed for rigorous reliability requirements of IEEE 802.3
- Squelch circuitry at all inputs rejects noise

SECTION 3

NETWORK EVALUATION BOARD


DP839EB
IEEE802.3
EVALUATION BOARD
NET 9804201-01 REV
S/N
J3
Reliability
QUALITY & RELIABILITY

DP839EB Network Evaluation Board Rev C

National Semiconductor Corp.

OVERVIEW

The National Semiconductor DP839EB Evaluation Board provides IBM PCs and PC Compatibles with Ethernet, Cheapernet and STARLAN® connections. The evaluation board is compatible with the PC-bus and requires only a ½ Size Slot for installation. The evaluation board utilizes National Semiconductor's Ethernet/Cheapernet chipset consisting of the DP8390 Network Interface Controller, the DP8391 Serial Network Interface and the DP8392 Coaxial Transceiver Interface. The DMA capabilities of the DP8390, coupled with 8 kbytes of buffer RAM, allow the Network Interface Adapter to appear as a standard I/O port to the system.

HARDWARE FEATURES

- Half-Size IBM PC I/O Card Form Factor
- DP8390 Network Interface Controller with DMA
- 8 kbyte on-board Multipacket Buffer
- Clean DMA Interface to IBM-PC
- Ethernet Interface via 15-Pin D Connector
- Cheapernet Interface via BNC Connector
- Starlan Support with Optional Daughter Card and 8-Pin Modular Phone Jack
- DP8391 Serial Network Interface
- DP8392 Coaxial Transceiver Interface (For Cheapernet)
- Low Power Requirement

SOFTWARE FEATURES

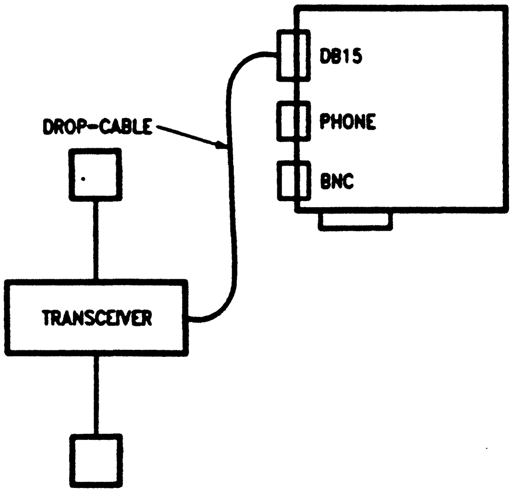
- No Software changes for conversion between Ethernet/Cheapernet and STARLAN
- Demonstration and diagnostic software available

NETWORK INTERFACE OPTIONS

The evaluation board supports three physical layer options: Ethernet, Cheapernet and STARLAN. When using Ethernet, a drop cable is connected to an external transceiver which is connected to a standard Ethernet network. (See *Figure 1*). When using Cheapernet, a low cost version of Ethernet, a transceiver is available on-board allowing direct connection to the network via the evaluation board. (See *Figure 2*). When using a STARLAN network, an optional daughter card replaces the SNI function and implements the required electronics to interface the DP8390 NIC to STARLAN. This configuration is illustrated in *Figure 3*. No software changes are needed for conversion between any of the described configurations.

HARDWARE DESCRIPTION

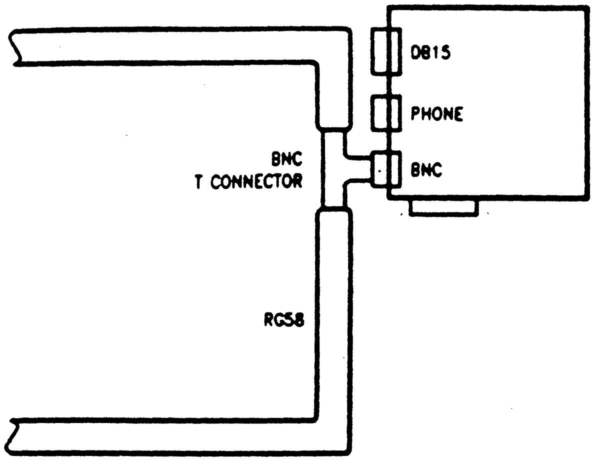
The block diagram shown in *Figure 4* illustrates the architecture of the Network Interface Adapter. The system/network interface is partitioned at the DP8390 Network Interface Controller (NIC). The NIC acts as both a master and a slave on the local bus. During reception or transmission of packets, the NIC is a master. When accessed by the PC, the NIC becomes a slave. The NIC utilizes a local 8-bit data bus connected to an 8k x 8 Static RAM for packet storage. The 8k x 8 RAM is partitioned into a transmit buffer and a receive buffer. All outgoing packets are first assembled in the packet buffer and then transmitted by the NIC. All incoming packets are placed in the packet buffer by the NIC and then transferred to the PC's memory. The transfer of data between the evaluation board and the PC is accomplished using the PC's DMA in conjunction with the NIC's Remote DMA. Two LS374 latches implement a bidirectional I/O port with the PC bus. The 8-bit transceiver (LS245) allows the PC



TL/F/9179-1

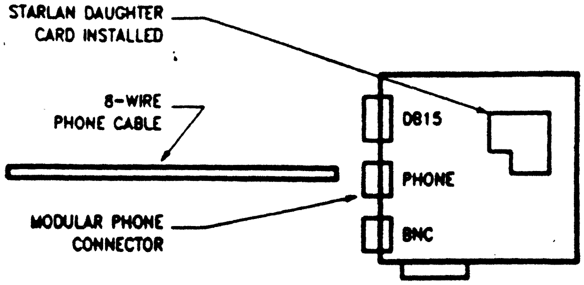
FIGURE 1. Ethernet Connection

IBM is a registered trademark of International Business Machines Corporation.
STARLAN® is a registered trademark of AT&T



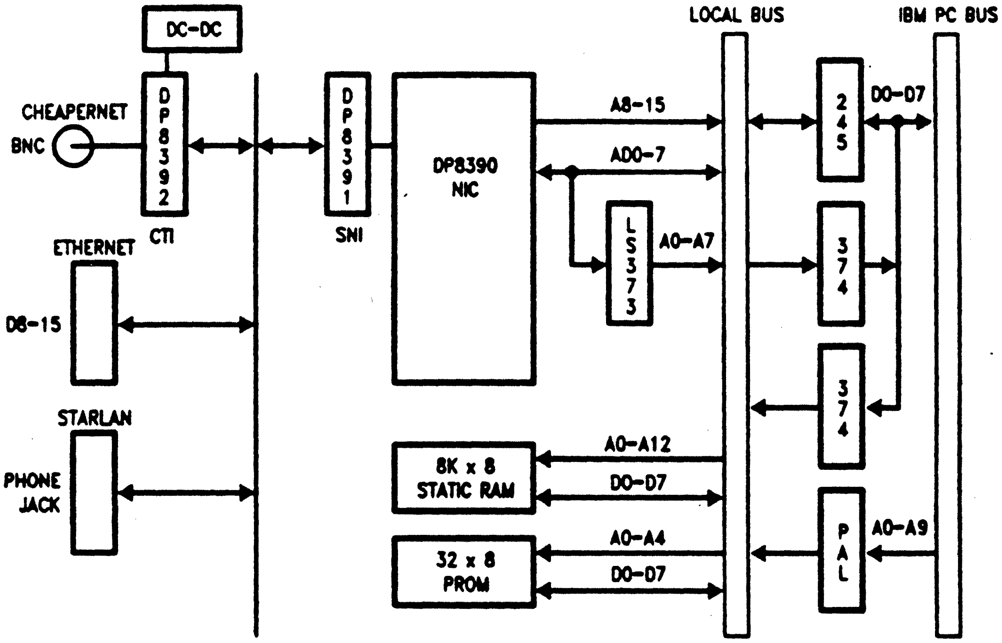
TL/F/9179-2

FIGURE 2. Cheapernet Connector



TL/F/9179-3

FIGURE 3. STARLAN Connector



TL/F/9179-4

FIGURE 4

to access to the NIC's internal registers for programming. A 32 x 8 PROM located on the evaluation board contains the unique Physical Address assigned to each board.

Since the NIC is accessing 8-bit memory, only a single demultiplexing latch is required for the lower 8-bits of address. An LS373 is provided for this purpose.

A 20L8 PAL provides the address decoding and support for DMA handshaking and wait state generation.

SOFTWARE SUPPORT

The evaluation board provides a simple programming interface for development of software. Several software packages are provided for evaluation and development of networks using the DP8390 Chip set. SDEMO is a demonstration program that provides a low level interface to the DP8390 NIC for transmission and reception of packets. SDEMO supports register dumps and simple register modification. CONF is a conferencing program which supports simple message transfer. WORKSTAT and SERVER support file transfer betw əen two nodes, one configured as a server and a second configured as a workstation. NLS, Network Load Simulator, is a program that simulates network loads based on statistical distributions of packet sizes, bursts and intervals. NLS is useful for performance measurement and debug of software drivers. NES, Network Evaluation Software, consists of sample software drivers implementing a low level interface to the evaluation board.

LOCAL MEMORY MAP

The DP8390 NIC accesses an 8k x 8 buffer RAM located in its 64 kbyte memory space. This buffer RAM is used for temporary storage of receive and transmit packets. Data from this RAM is transferred between the host (the PC) and the evaluation board using the DP8390 NIC's remote DMA channel. An ID address PROM, containing the physical address of the evaluation board is also mapped into the memory space of the NIC.

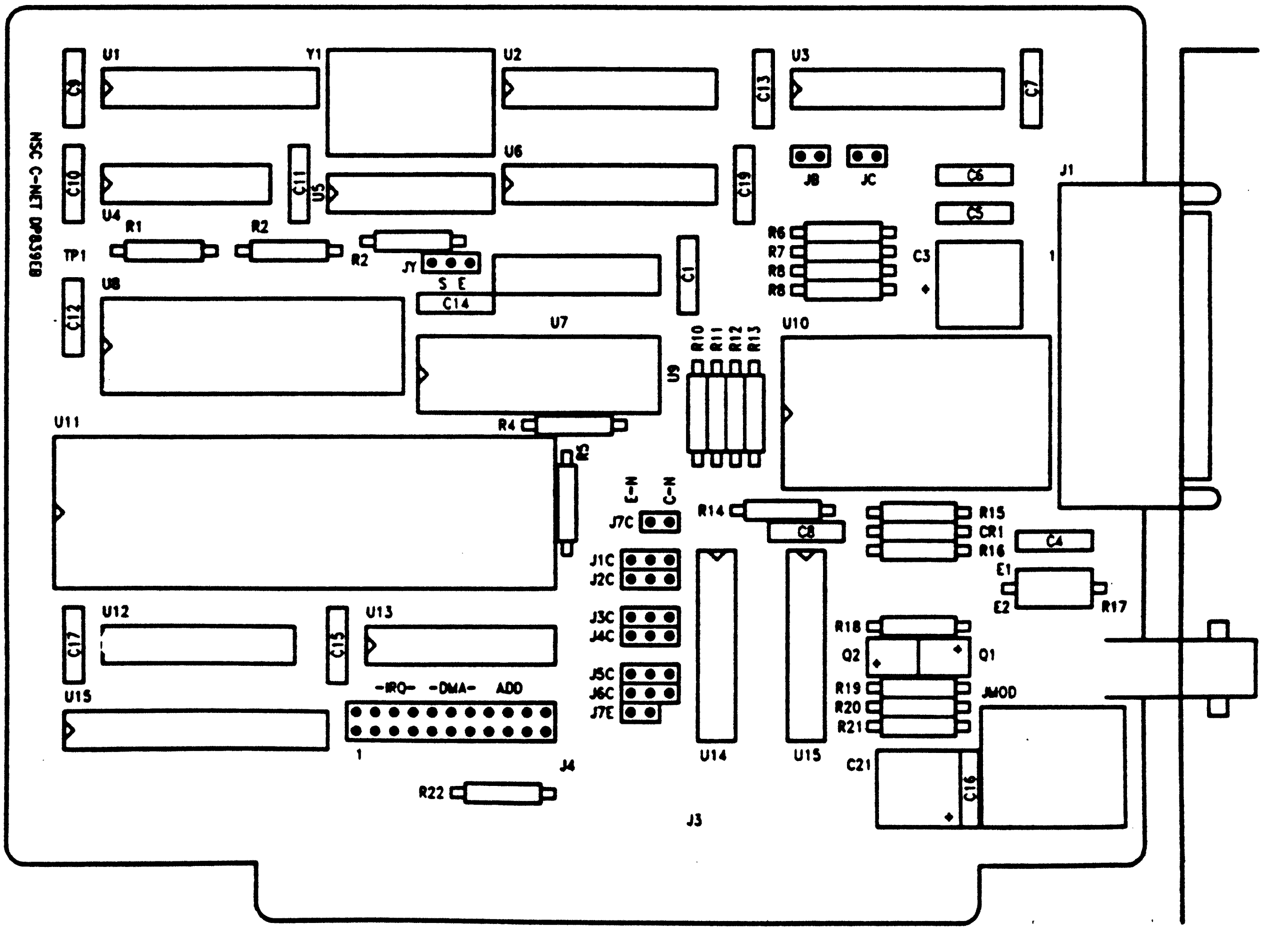
Note: Partial decoding is performed on the PROM and RAM which will result in these devices appearing at other locations in the 64k memory space. The first occurrance of the PROM and RAM are used for programming purposes.

0000h 001fh	PROM
• •	• •
2000h 3fffh	8k x 8 BUFFER RAM
• •	• •
ffffh	•

PROM FORMAT

Each evaluation board is assigned a unique network (physical) address. This address is stored in a 74S288 32 x 8 PROM. The physical address is followed by a checksum. The checksum is calculated by exclusive OR-ing the 6 address bytes with each other. At initialization the software reads the PROM, verifies the checksum and loads the NIC's physical address registers. The following format is used in the PROM:

Address	Contents
00h	ADDRESS 0 (Physical Address Most Significant Byte)
01h	ADDRESS 1
02h	ADDRESS 2
03h	ADDRESS 3
04h	ADDRESS 4
05h	ADDRESS 5 (Physical Address Least Significant Byte)
06h	CHECKSUM (XOR OF ADDRESS 0-5) OPTIONAL
07h	REV. NUMBER
08h	MANUFACTURE LOT #
09h	MANUFACTURE DATE (MONTH)
10h	MANUFACTURE DATE (YEAR)
11h–1fh	RESERVED



TL/F/9179-5

FIGURE 5

I/O SPACE

The I/O space and Ethernet/Cheapernet configurations are selected using the various I/O jumpers. There are 4 sets of jumpers that should be programmed prior to installation of the evaluation board into the PC environment. There are:

J4	I/O address, interrupt selection, DMA channel assignment
J1C-J7C, J7E	Select Ethernet or Cheapernet
JY	Selects Ethernet/Cheapernet or STARLAN clocking
JB, JC	Selects B3 or C stepping silicon

Figure 5 depicts the location of the jumpers on the evaluation board.

The Factory Installed Configuration Is:

J4	I/O base = 300h
	Interrupt = IRQ3
	DMA = DREQ1, DACK1

J1C-J7C, J7E	Cheapernet selected
JY	Ethernet/Cheapernet clock selected
JB, JC	JB shorted selects revision B silicon, JC selects REV C

I/O SPACE

The evaluation board uses 32 I/O locations in the PC's I/O space. The base address IS FIXED AT 300H AND IS NOT SELECTABLE using jumpers. (See Switch settings section.) The I/O map is shown below:

BASE + 00h	COMMAND REGISTER
01h	NIC REGISTER
02h	SPACE
03h	•
04h	•
05h	•
06h	•
07h	•
•	•
0fh	•
10h	I/O PORTS
•	•
1f h	•

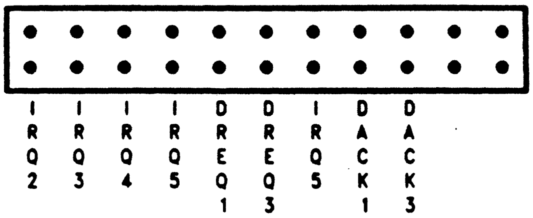
NOTES:

The NIC's Command Register is always mapped at Base + 0. The NIC registers are Base + 01 to Base + 0f will contain different registers depending on the value of bits PS0 and PS1 in the Command Register. These two bits select one of four register pages. For additional information consult the DP8390 data sheet.

The NIC uses a DMA channel to read/write data from/to the 8k x 8 Buffer RAM on the evaluation board. Typically a DMA channel on the PC is used in conjunction with the NIC's remote DMA. The I/O ports are then serviced by the DMA channel. If a DMA channel on the PC is not available, the NIC's DMA can still be used by accessing the I/O ports using programmed I/O. Reading the I/O port address will result in a $\overline{\text{RACK}}$ strobe to the NIC while writing the I/O port address will result in a $\overline{\text{WACK}}$ strobe to the NIC.

SWITCH SETTINGS

Jumper J4 allows assignment of DMA channel assignments and Interrupt Request assignments. The jumper configuration is shown below and described in the following sections.



TL/F/9179-6

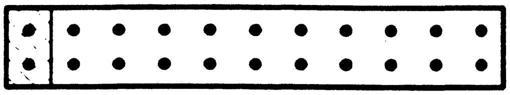
I/O BASE ADDRESS

THE I/O BASE FOR REV C BOARDS IS FIXED AT 300H AND IS NOT SELECTABLE.

INTERRUPTS

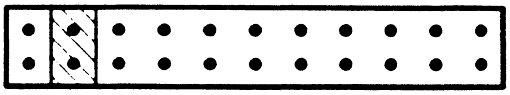
The NIC will generate interrupts based on received and transmitted packets, completion of DMA and other internal events. The interrupt can be connected to Interrupts 2, 3, 4 or 5 (IRQ 2, 3, 4, 5) via Jumper J4. Interrupt 5 is also provided as a software driven DMA Channel. If Interrupt 5 is being used as a DMA channel Interrupt 5 cannot be chosen for the NIC interrupt. The figures below illustrate the jumper positions for the various interrupt levels.

Interrupt 2



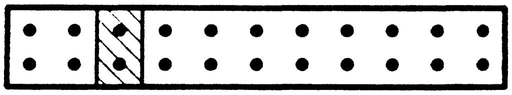
TL/F/9179-9

Interrupt 3
(Factory Installed)



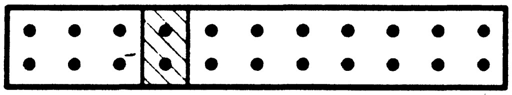
TL/F/9179-10

Interrupt 4



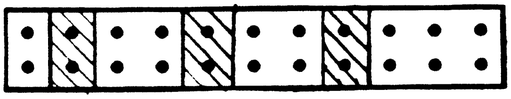
TL/F/9179-11

Interrupt 5



Note: Rev C demo software will not work unless the factory configuration for jumper block J4 is used.

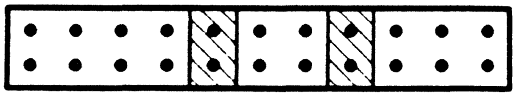
Factory Configuration:



DMA

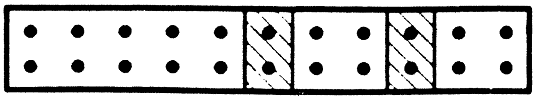
The evaluation board requires 1 DMA channel on the PC expansion bus, DMA channel 1 or 3 can be selected. The corresponding DACK line must also be installed on Jumper J4. If your PC contains an SDLC card then DMA channel 3 must be selected.

DMA Channel 1
(Factory Installed)



TL/F/9179-15

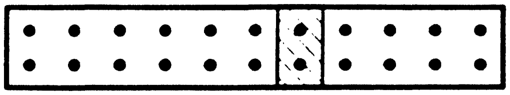
DMA Channel 3



TL/F/9179-16

If a DMA channel is not available an interrupt driven routine can be used to move data between the PC and the buffer memory on the evaluation board. Interrupt 5 is used for this function.

IRQ 5 for DMA

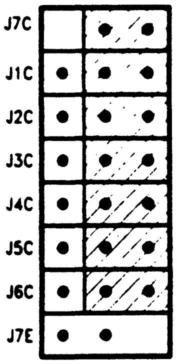


TL/F/9179-17

SELECTING ETHERNET OR CHEAPERNET

Two 10Mbit/sec Interface option are available, a connection to an external transceiver via the DB-15 connector, or a direct interface to a BNC T-connector. Seven jumpers are used to select the appropriate option. These jumpers are labeled J1C-J7C and J7E.

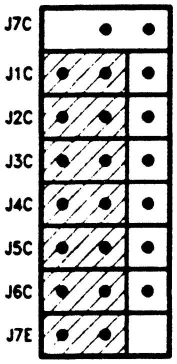
For Cheapernet the following jumpers should be shorted:



TL/F/9179-18

(Factory Installed)

For Ethernet the following jumpers should be shorted.



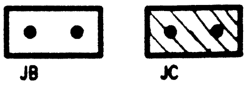
TL/F/9179-19

Double check the jumper positions prior to powering up the board.

SILICON VERSIONS

The evaluation board supports two revisions of silicon. JB selects REV B3 silicon. J selects REV C silicon.

(JC Factory Installed)

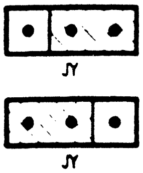


TL/F/9179-20

OSCILLATOR

For future STARLAN daughter board applications, a divide by 10 scaler is provided to allow switching of the bus clock on the DP8390 to 2 MHz. The jumper, labeled JY should be configured as shown.

Ethernet, Cheapernet
(Factory Installed)



TL/F/9179-21

APPENDICES

The remainder of this document contains the evaluation board parts list, schematic and PAL descriptions.

PARTS LIST

Part No.	Description	Quantity
U1	74LS373N	1
U2,U6	74LS374N	2
U3	74LS245N	1
U4	74S288	1
U5	74LS290N	1
U7	74HC74	1
U8	HM6264-100 or Eqv.	1
U9	DP8391	1
U10	2VP5U9	1
U11	DP8390	1
U12		
U13	PAL16R4	1
U14	PE64103	1
U15	DP8392	1
U16	PAL20L8	1
Y1	Crystal Oscillator Module 20 MHz +/−.01%	1
Q1,Q2	PN200-T0-92 pkg.	2
CR1	1N914 diode	1
R1,R2,R3,R22	4.7K 5% carbon	4
R6,R7,R8,R9	39Ω 1%	4
R4,R5	510Ω 5% carbon	2
R10,R11,R12,R13	1.5K 5% carbon	4
R14	1.0k 1%	1
R17	1 MΩ ½W carbon	1
R19	1.2k 5% carbon	1
R20	3.6k 5% carbon	1
R21	430Ω 5% carbon	1
R15	Shorted	1
R18	RRX1-TDB Open	1
R16	RRX2-TBD Shorted	1
C1, C7–C17 (C10, C19 optional)	0.47 μF	13
C3, C21	47 μF ELECTROLYTIC	1
C5, C6	0.01 μF Ceramic	2
C4	0.01 μF Ceramic (600V)	1
JUMPERS	0.03 sp clips	16/board
J2	PANEL MT. BNC	1
J1	DB15	1
48-PIN SOLDERTAIL SOCKET	for U11	1
24-PIN SOLDERTAIL SOCKET	for U16	1
24-PIN AUGAT SOCKET	for U9	1
20-PIN SOLDERTAIL SOCKET	for U12, U13	2
14-PIN SOLDERTAIL SOCKET	for U7	1

Note: Q1, Q2, R21, R20, R19 not required for DP8392-4 parts or later. R15, R18, R16 provided as an option to drive 93Ω coax.

```
PAL20L8                                        National Semiconductor
I/O Address Decode Plus for NIA                January 14, 1986
DECODE                                         Doug Wilson

     A8    A7    A6    A5    A4    NC  /NMRD  NA13 /IORD  /IOWR GND
PRQ /DACK /WAIT AEN /RACK /WACK /CSX /CSN  NC  /CSROM /ACK  VCC

CSN =   /AEN* A9* A8* /A7* /A6* /A5* /A4* IOWR +
        /AEN* A9* A8* /A7* /A6* /A5* /A4* IORD

RACK = /AEN* A9* A8* /A7* /A6* /A5* A4* PRQ* IORD +
        DACK*  IORD

WACK = /AEN* A9* A8* /A7* /A6* /A5* A4* PRQ* IOWR +
        DACK*  IOWR

CSX =    CSN +
       /AEN* A9* A8* /A7* /A6* /A5* A4* IORD +
       /AEN* A9* A8* /A7* /A6* /A5* A4* IOWR

IF (CSX)
WAIT = /ACK *  CSN  +
       /PRQ * /CSN

CSROM = /NA13 * NMRD
```

Description

This pal performs the I/O decodes for selecting the NIC, and the handshake signals for NIC's remote dma. The pal supports the dma channels of the PC for remote DMA transfers with the NIC and also allows the use of string I/O between 80286 PC's and NIC's remote DMA.

Using DECODE fixes the I/O BASE of the card at 300h. NIC registers fall in the space 300h - 30fh. To use the string I/O port, reads and writes are done to port 310h.

Wait states are inserted (WAIT) to the PC bus when register accesses are given and the NIC is busy performing other operations (such as local bursts). When the NIC is ready, /ACK is given and no (more) wait states are inserted.

Wait states may also be inserted during remote DMA operations and 80286 machines using string I/O's. WAIT occurs during a remote read if the PC AT's /IORD goes low before the DP8390's PRQ goes high. Similarly, WAIT occurs during a remote write if the PC AT's /IOWR goes low before the NIC's PRQ goes high.

NIC registers are accessed when CSN (Chip Select NIC) is asserted. The IORD and IOWR terms are included to ensure that the address lines are valid when CSN is given.

The RACK and WACK signals are used by the NIC's remote DMA channel to acknowledge the end of a single read or write operation through the remote DMA I/O ports. These port are addressable by the PC DMA channel with DACK and IORD or IOWR, or by addressing the I/O location 310h (with string I/O's).

CSX is used to enable the TRI-STATE output of WAIT during a register access (CSN), and during string I/O to the remote DMA's I/O port (CSX).

CSROM provides address decode for the card's address PROM. The card's unique Ethernet address is transferred to the system using the NIC's remote DMA.

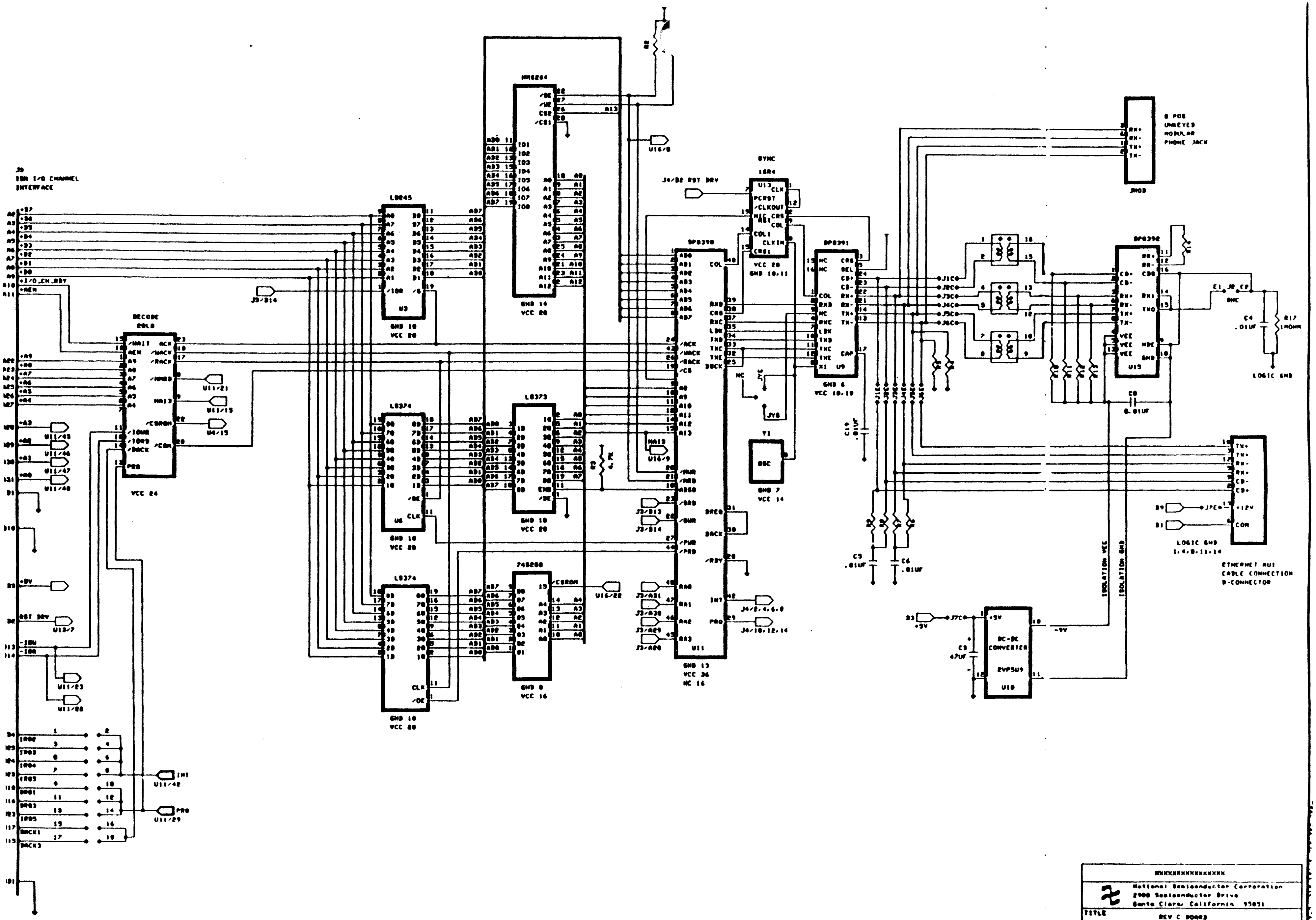
```
PAL16R4                                          National Semiconductor
   inite Transmit Defer Fix                      January 14, 1987
SYNC                                             Doug Wilson


CLK CRS NC /MWRIN NC NC PCRST CLKIN COL GND
/ENABLE /CLKOUT /MWROUT COL1 CRS1 CRS2 NC NC /NICRST VCC

CLKOUT = CLKIN

/CRS1 := /CRS

/CRS2 := /CRS1 +
         /CRS

/COL1 := /COL

NICRST = PCRST

MWROUT = MWRIN
```

Description

This pal synchronizes CRS and COL signals from the SNI to ensure that NIC never sees half-level voltages on CRS or COL. This is achieved by sampling them with an inverted 20 Mhz clock (CLKOUT) and giving the COL1 and CRS1 signals to the NIC.

NICRST is a power-on reset signal given to the NIC, generated by inverting the PC signal RST DRV.

MWROUT just passes MWRIN thru. This exists to provided a jumper for the MWR signal that was needed for the SCAP4 pal (Rev B boards).



National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, California 95051

TITLE
REV C BOARD
SUPPORTS STRING I/O AND DMA

ENGINEER DATE DOUG WILSON 408-721-3427	DESIGN SITE
APPROVED DATE	RELEASED DATE
REVISION C	DATE 2/10/87
SHEET NO. 1	NO. OF SHEETS 1

National
Semiconductor
Corporation

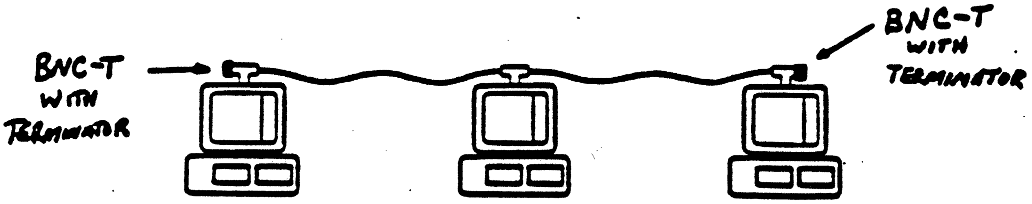
Cable Installation and Ordering Guide
Preliminary
July 1986

Cable Installation and Ordering Guide

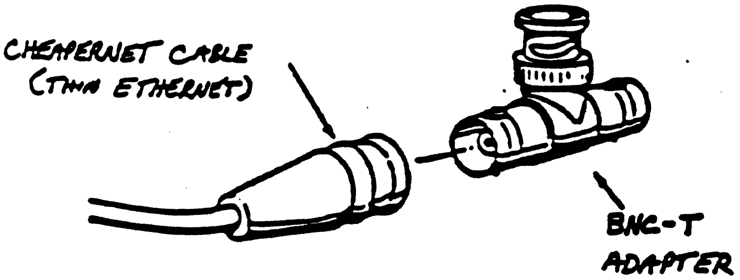
Once the NIAs have been installed into the PCs (see the NIA Hardware Reference Manual for installing NIAs) they are ready to be cabled together.

CHEAPERNET

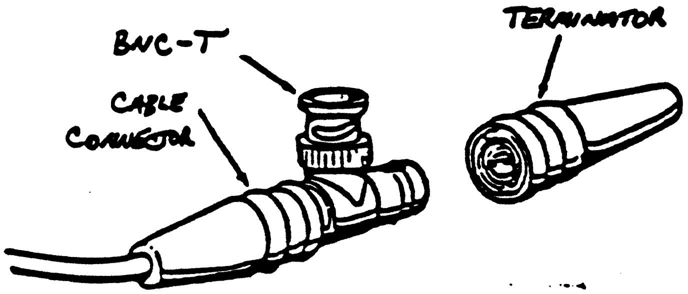
The connection scheme shown below is for Cheapernet (thin Ethernet) networks.



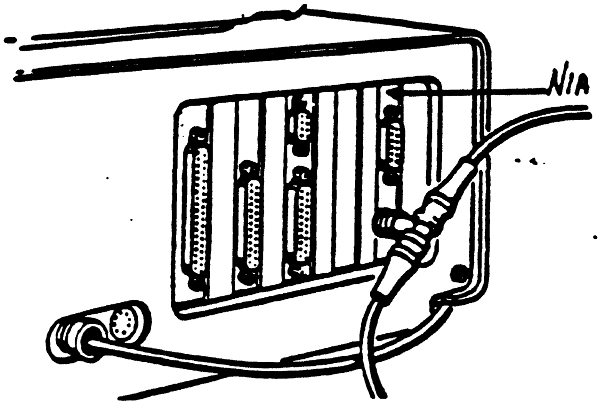
The network is constructed by first connecting two lengths of cable together with a BNC-T adapter as shown below.



At each end of the network a BNC-T adapter and terminator are connected as depicted below.

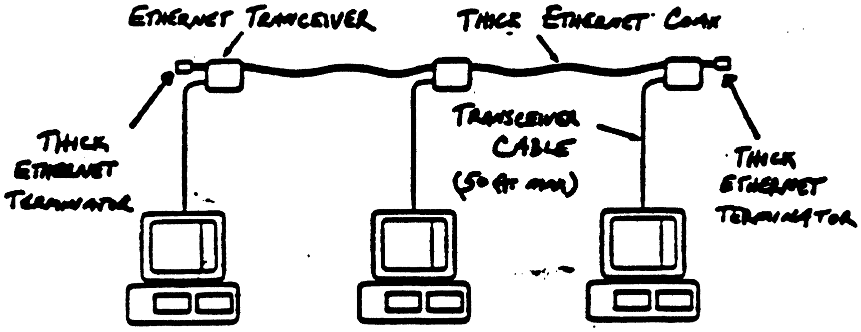


Finally the connection to the NIA board is made with the BNC-T as shown below.

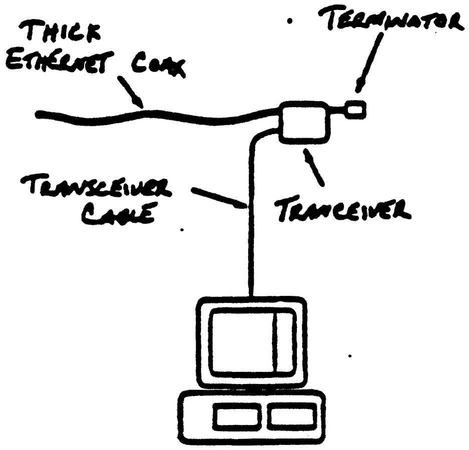


ETHERNET

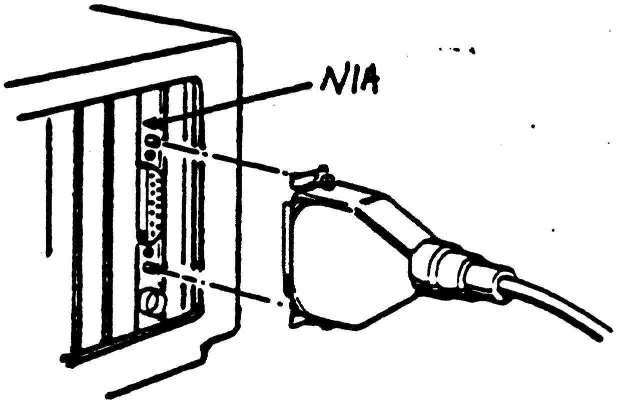
The connection scheme shown below is for Ethernet (thick Ethernet) networks.


ETHERNET TRANCEIVER
THICK ETHERNET COAX
THICK ETHERNET TERMINATOR
TRANSCEIVER CABLE (50 ft max)
THICK ETHERNET TERMINATOR

The network is constructed by first connecting two lengths of thick Ethernet coax together with an Ethernet transceiver box. Transceiver boxes are also added at each end of the coax. A terminator must be added to the ends of the thick Ethernet segment as depicted below.



Finally the connection to the NIA board is made with the transceiver cable as shown below.



CONNECTING A TRANCEIVER CABLE AND NIA

CABLE PROCUREMENT

Both Cheapernet (thin Ethernet) and thick Ethernet cableing and accessories may be ordered through INMAC. Included is a copy of the LAN products offered by INMAC in the June 1986 INMAC catalog. Also included is INMAC ordering information.

SECTION 4

SOFTWARE AND SAMPLE DRIVER EXAMPLES

Software Reference Manual

- Computer Conferencing Program
- Demonstration Network Software
- NIA Access Software
- Network Load Simulator
- Network Evaluation Software
- Writing Drivers for the DP8390

National
Semiconductor
Corporation

Software Disclaimer
Preliminary
July 1986

SOFTWARE DISCLAIMER

The enclosed software is not a supported product. It was developed and supplied solely for demonstration purposes. Although the demonstration software has been thoroughly tested, National Semiconductor does not assume responsibility for any defects or anomalies found in the software. Source code has been provided to allow modification or troubleshooting by the customer if necessary. Any modification of the software is the sole responsibility of the customer.

SOFTWARE UPDATE FOR THE DP8390

The enclosed diskette contains software upgrades for using DP8390 Revision C silicon. All of the programs provided in the original DP839EB evaluation kit have been upgraded. The operation of all the upgraded software is the same as in the original DP839EB software.

The program name changes from the original DP839EB versions are listed below:

Rev. B3+	—>	Rev. C
Sdemo	--->	Sdemoc
Conf	--->	Confc
Workstat	--->	Workstatc
Server	--->	Serverc
Nes	--->	Nesc300
NLS	--->	NLSFF

National
Semiconductor
Corporation

Computer Conferencing Program

CONF.EXE . Preliminary
July 1986

OVERVIEW

The Computer Conferencing Program (CONF) provides an introduction to local area networks (LAN). Using National Semiconductor's Network Interface Adapter (NIA). CONF sets up a real-time interactive networking environment. Messages entered at one terminal propagate throughout the network and are displayed at the companion NIA equipped terminal running CONF. The NIA is an IBM PC-compatible LAN demonstration board designed to evaluate National's IEEE 802.3-compatible DP8390 network chip set. The DP8390 chip set is one of the latest of National's Advanced Peripheral Processing Solutions (APPS:) family of VLSI circuits for microprocessor peripheral interface applications. The board plugs into any PC-compatible machine and incorporates all of the components required to provide a LAN interface to Ethernet or Cheapernet networks. An optional daughter card allows LAN interface with low cost STARLAN networks.

FEATURES

- SIMPLE SETUP/CONFIGURATION
- MESSAGES TRANSFERRED AND DISPLAYED ON EACH TERMINAL
- SPLIT SCREEN DISPLAY WITH WINDOWS FOR TRANSMIT AND RECEIVE

REQUIREMENTS

CONF requires a network of at least two NIA equipped nodes. Each NIA must be previously configured as outlined in the Installation manual. The PC must be operating on DOS 3.0 or greater.

LOADING AND RUNNING

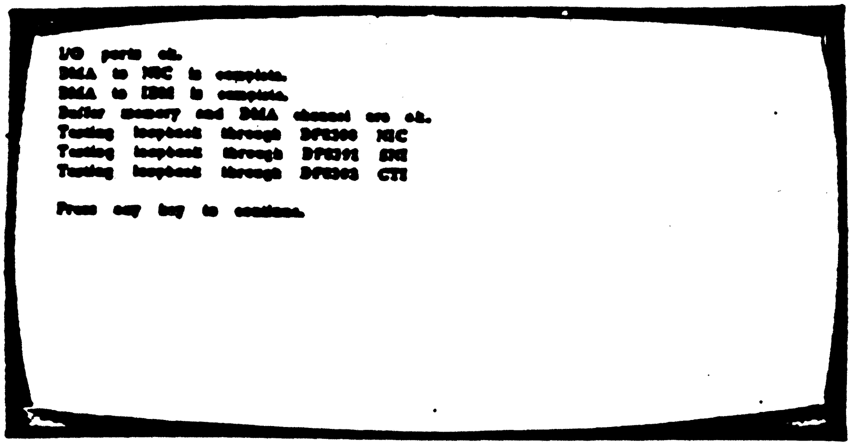
Boot the node using DOS 3.0 or greater. Insert the EXECUTABLE CODE disk into the active drive. Invoke CONF from the keyboard by typing CONF [return]. Note the status window which indicates the condition of the NIA (see figure 1). If an error message is present follow these troubleshooting procedures:

1) Powerdown and re-boot.

2) If an error occurs during I/O or DMA testing then check the jumper blocks on the NIA board (see the Network Interface Adapter Hardware Reference Manual).

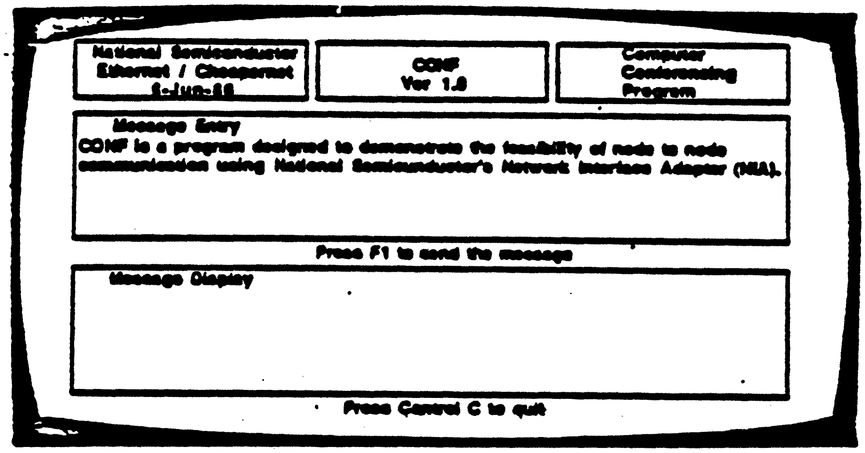
3) If any other error please check the network cable connections and terminators.

If no errors are indicated, press any key to exit the status window.



```
I/O ports ok.
DMA to NIC is complete.
DMA to IBM is complete.
Buffer memory and DMA channel are ok.
Testing loopback through DP8390 NIC
Testing loopback through DP8391 SNI
Testing loopback through DP8392 CTI

Press any key to continue.
```

Figure 1

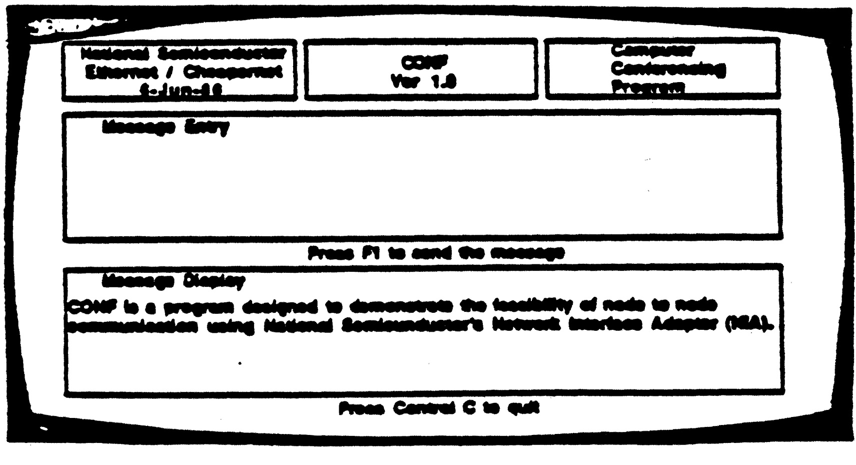


```
National Semiconductor        CONF          Computer
Ethernet / Cheapernet        Ver 1.0       Conferencing
     6-Jun-86                                Program

    Message Entry
CONF is a program designed to demonstrate the feasibility of node to node
communication using National Semiconductor's Network Interface Adapter (NIA).

                     Press F1 to send the message

    Message Display

                       Press Control C to quit
```

Figure 2

At this point CONF is ready to send and receive messages through the network (provided there is at least one other CONF equipped node on the network). To initiate communication enter a message from the keyboard. The message will appear on your terminal in the Message Entry window as depicted above (figure 2). To send the message simply press F1 (function key 1). The message will propagate through the network and appear in the Message Display window of any other node operating with CONF. The message is automatically sent when the Message Entry window fills to capacity while typing a message.

Messages from other nodes running CONF will appear in the Message Display compartment on your terminal (figure 3). These messages will be replaced by any subsequent transmissions from other nodes. To exit CONF press [control] C.



```
National Semiconductor        CONF          Computer
Ethernet / Cheapernet        Ver 1.0       Conferencing
     6-Jun-86                                Program

    Message Entry

                     Press F1 to send the message

    Message Display
CONF is a program designed to demonstrate the feasibility of node to node
communication using National Semiconductor's Network Interface Adapter (NIA).

                       Press Control C to quit
```

Figure 3

National
Semiconductor
Corporation

Demonstration Network Software
SERVER.EXE, WRKSTAT.EXE , Preliminary
July 1986

OVERVIEW

The Demonstration Network Software package (D-NET) demonstrates file transfer operations within local area networks (LAN). The D-NET software package contains two programs which together emulate the networking relationship between a workstation and a file server. SERVER is the file server emulator and shares its disk drive capabilities with workstations on the network. WRKSTAT is the workstation emulator which may access the file server's disk. Using National Semiconductor's Network Interface Adapter (NIA), D-NET sets up a real-time interactive networking environment in which the workstation can read directories, type and transfer files to and from the file server. The NIA is an IBM PC-compatible LAN demonstration board designed to evaluate National's IEEE 802.3-compatible DP8390 network chip set. The DP8390 chip set is one of the latest of National's Advanced Peripheral Processing Solutions (APPS:) family of VLSI circuits for microprocessor peripheral interface applications. The board plugs into any PC-compatible machine and incorporates all of the components required to provide a LAN interface to Ethernet or Cheapernet networks. An optional daughter card allows LAN interface with low cost STARLAN networks.

FEATURES

- SIMPLE SETUP/CONFIGURATION
- FILES TRANSFERRED BETWEEN EACH TERMINAL
- A COMPLETE MENU DRIVEN SOFTWARE PACKAGE

REQUIREMENTS

D-NET requires a network of at least two NIA equipped nodes. Each NIA must be previously configured as outlined in the Installation manual. Additionally, COMMAND.COM must reside in the root directory of the active disk drive. The PC must be operating on DOS 3.0 or greater.

LOADING AND RUNNING

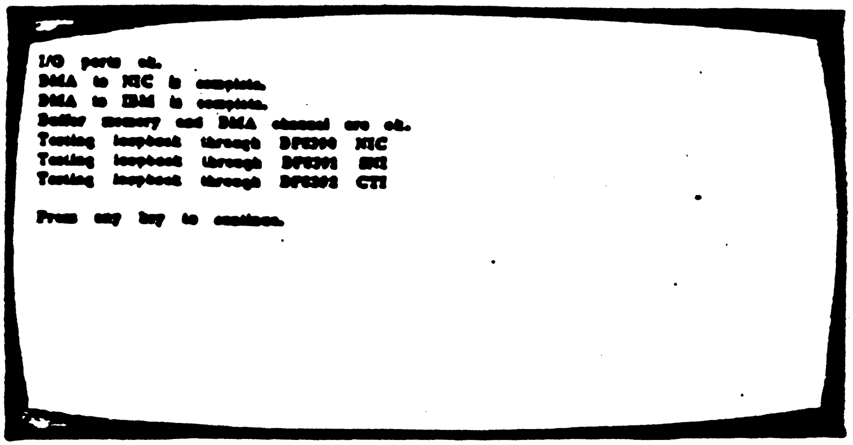
Boot each node using DOS 3.0 or greater. Insert the EXECUTABLE CODE disks into the active drive of each PC. Invoke SERVER from one keyboard by typing SERVER [return]. Type WRKSTAT [return] on the other. Note the status window on each terminal which indicate the conditions of each NIA (see figure 1). If an error message is present follow these troubleshooting procedures:

1) Powerdown and re-boot.

2) If an error occurs during I/O or DMA testing then check the jumper blocks on the NIA board (see the Network Interface Adapter Hardware Reference Manual).

3) If any other error please check the network cable connections and terminators.

If no errors are indicated, press any key to exit the status window.



```
I/O ports ok.
DMA to NIC is complete.
DMA to IBM is complete.
Buffer memory and DMA channel are ok.
Testing loopback through DP8390 NIC
Testing loopback through DP8391 SNI
Testing loopback through DP8392 CTI

Press any key to continue.
```

Figure 1

THE FILE SERVER

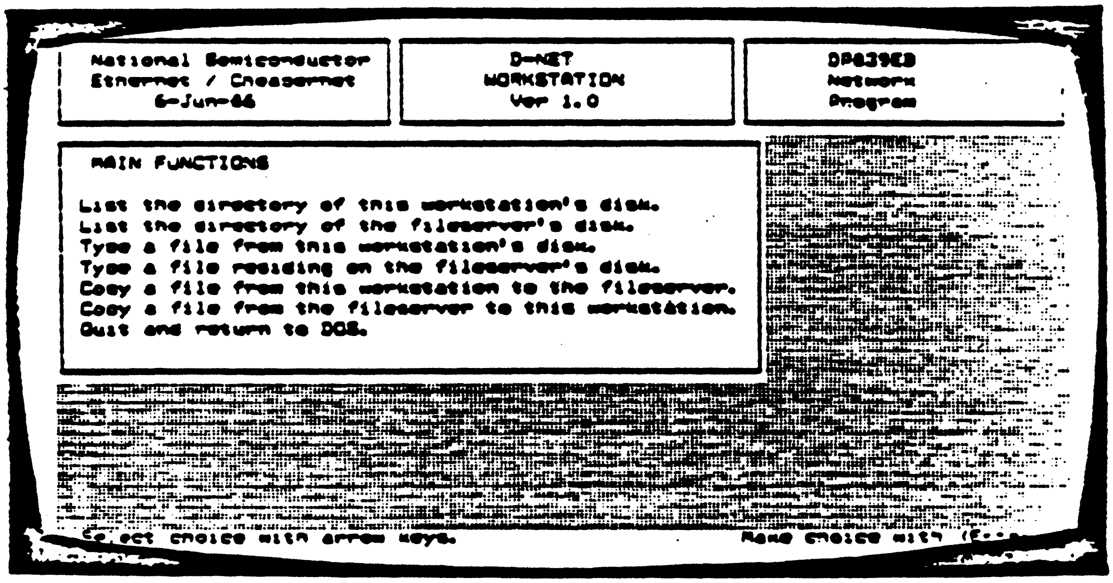
The file server must have a non write-protected disk inserted in its active drive (or a fixed disk as an active drive). The system will not work properly if the file server cannot read and write onto a disk. The file server is now ready to interact with the workstation.

THE WORKSTATION

A menu of WRKSTAT's main functions is displayed on the terminal as shown in figure 2. The items on the menu are accessed using the cursor control keys of the PC's numeric keypad. Pressing the [2↓] key moves the cursor down the menu, and the [8↑] key moves it up. If the cursor control keys are not performing these functions press the [NumLock] key. The [return] key is used to select the desired function.

Files must be 20Kbytes or less when typing or copying.

When typing a file use [control][S] to stop/start scrolling or [control][C] to abort typing.



```
National Semiconductor        D-NET            DP839EB
Ethernet / Cheapernet       WORKSTATION        Network
      6-Jun-86               Ver 1.0           Program

MAIN FUNCTIONS

List the directory of this workstation's disk.
List the directory of the fileserver's disk.
Type a file from this workstation's disk.
Type a file residing on the fileserver's disk.
Copy a file from this workstation to the fileserver.
Copy a file from the fileserver to this workstation.
Quit and return to DOS.

Select choice with arrow keys.                Make choice with (F..
```

Figure 2

National
Semiconductor
Corporation

NIA Access Software
SDEMO.EXE Preliminary
June 1986

OVERVIEW

The Network Interface Adapter Access Software package (henceforth referred to as SDEMO) is a demonstration/learning tool which provides the means to investigate the NIA's capabilities. Using National Semiconductor's Network Interface Adapter (NIA), SDEMO sets up a real-time interactive networking environment. The NIA is an IBM PC-compatible LAN demonstration board designed to evaluate National's IEEE 802.3-compatible DP8390 network chip set. The DP8390 chip set is one of the latest of National's Advanced Peripheral Processing Solutions (APPS:) family of VLSI circuits for microprocessor peripheral interface applications. The board plugs into any PC-compatible machine and incorporates all of the components required to provide a LAN interface to Ethernet or Cheapernet networks. An optional daughter card allows LAN interface with low cost STARLAN networks.

FEATURES

- SIMPLE SETUP/CONFIGURATION
- COMPLETE NIC ACCESS AND CONTROL
- A COMPLETE MENU DRIVEN SOFTWARE PACKAGE

REQUIREMENTS

SDEMO requires an NIA equipped PC operating with DOS 3.0 or greater. The NIA must be previously configured as outlined in the NIA Hardware Reference Manual. Additionally, the program storage disk should be conditioned using the Installation Utility.

LOADING AND RUNNING

Boot the PC with DOS 3.0 or greater. Invoke SDEMO from the keyboard by typing SDEMO [return]. The terminal will now display the Main Functions selection screen as shown in figure 1.

ACCESSING THE MENUS

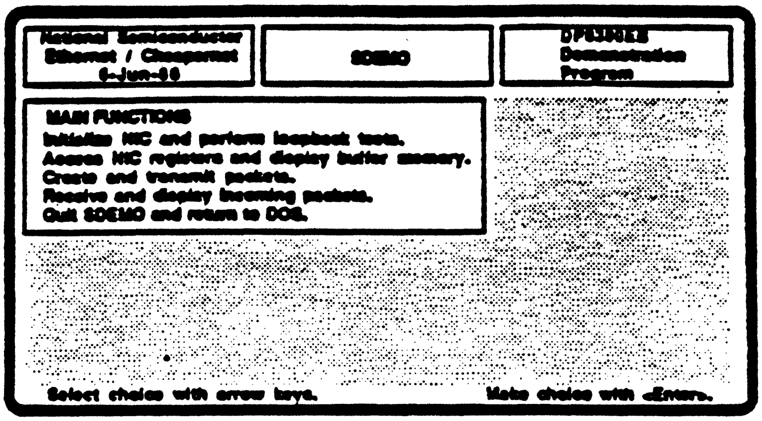
The items on the menu are accessed using the cursor control keys of the PC's numeric keypad. Pressing the [2↓] key moves the cursor down the menu. [8↑] moves it up. If the cursor control keys are not performing these functions press the [NumLock] key. The [return] key is used to select the desired function. The Escape key [esc] is used by SDEMO to exit the current window.

Throughout this document the various windows will be numerically referred to by their corresponding section headings. It should be noted that these numbers exist only in this document and are not displayed on the terminal by SDEMO.

Main Functions

The Main Functions window is your viewport into SDEMO's four main functions of initialization, register and memory access, packet transmission and packet reception.

SDEMO does not initialize the NIA when it is invoked. This feature is useful for determining the state that the NIA has been left in by another process (a system crash, perhaps). To use SDEMO for anything other than determining the current state of the NIA it is necessary to select the initialization option (window 1.1).

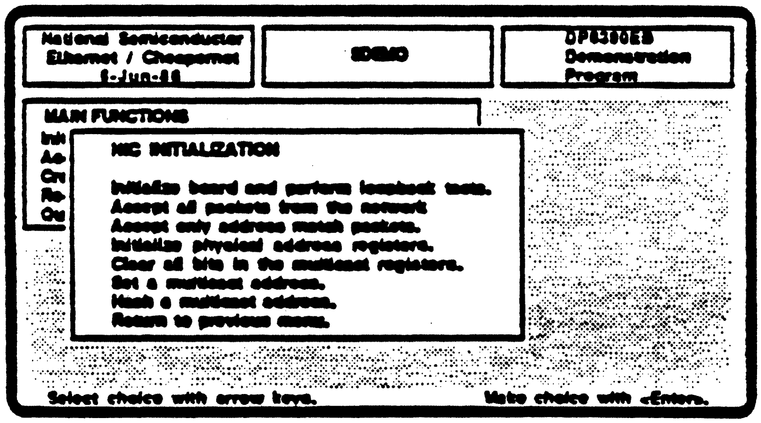


```
National Semiconductor                         DP8390EB
 Ethernet / Cheapernet        SDEMO          Demonstration
       6-Jun-88                                 Program

 MAIN FUNCTIONS
 Initialize NIC and perform loopback tests.
 Access NIC registers and display buffer memory.
 Create and transmit packets.
 Receive and display incoming packets.
 Quit SDEMO and return to DOS.

 Select choice with arrow keys.              Make choice with <Enter>.
```

Main Functions

1. Initiaiize NIC and perform loopback tests.

This window (figure 1.0) becomes incumbent when the initialize option is selected from the Main Functions window. The options in this window are for power on initialization and for receive filter initialization.



```
National Semiconductor                         DP8390EB
 Ethernet / Cheapernet        SDEMO          Demonstration
       6-Jun-88                                 Program

 MAIN FUNCTIONS
 Init   NIC INITIALIZATION
 Ac
 Cr     Initialize board and perform loopback tests.
 Re     Accept all packets from the network
 Qu     Accept only address match packets.
        Initialize physical address registers.
        Clear all bits in the multicast registers.
        Set a multicast address.
        Hash a multicast address.
        Return to previous menu.

 Select choice with arrow keys.              Make choice with <Enter>.
```

Figure 1.0

1.1. Initialize board and perform loopback tests.

This is the power on initialization option and must be selected before any other options can be executed. This option need only be selected once for each SDEMO session. After selecting this option, note the status window which indicates the condition of the NIA (see figure x). If an error message is present follow these troubleshooting procedures:

1) Powerdown and re-boot.

2) If an error occurs during I/O or DMA testing then check the jumper blocks on the NIA board (see the Network Interface Adapter Hardware Reference Manual).

3) If any other error please check network cable connections.

Press any key to exit the status window.

1.2. Accept all packets from the network.

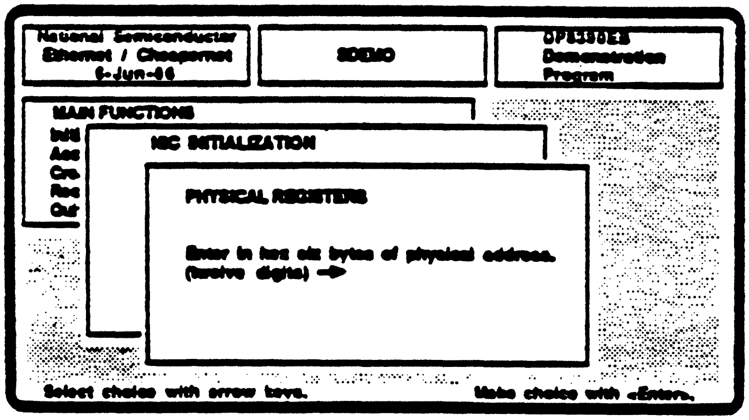
This selection puts the NIC into promiscuous mode. The NIC's Receive Configuration Register is programmed to 1Fh and all multicast bits are set. Any packet on the network will be accepted.

1.3. Accept only address match packets.

In this mode only packets which have a destination address that matches the address residing in the Physical Address Registers and packets that have a multicast address that hashes to a bit set in the Multicast Address Registers will be accepted. The Physical Address Registers must be initialized using window 1.4 before the address match filter will work.

1.4. Initialize physical address registers.

This option opens a window (figure 1.4) which prompts for the physical address that is to be programmed into the Physical Address Registers. Twelve hex digits are required. Valid hex digits are 0 → 9, and A → F. The registers are automatically programmed with the entered address.



Enter in hex six bytes of physical address.
(twelve digits) →

Figure 1.4

1.5. Clear all bits in the multicast registers.

This option clears all the Multicast Address Register bits to zero so that no multicast address packets will be recognized. This option should be used before any multicast addresses are set to insure that only the desired multicast addresses are recognized.

1.6. Set a multicast address.

This option opens a window which prompts for twelve hex digits of the multicast address. Valid hex digits are 0 → 9, and A → F. Once entered, the multicast address is hashed by a routine similar to CRC generation (see PD8390 Data Sheet for more information on multicast address hashing). After the multicast address filter bit number is found by the hashing routine it is displayed and the bit is set in the Multicast Address Registers. The desired multicast address packet will now be recognized by the receive filters.

1.7. Hash a multicast address.

Similar to 1.6 except the Multicast Address Register bit is not set. The option only determines which bit a multicast address hashes to. The multicast address will not be recognized by the receive filters unless it has been set using window 1.6.

1.8. Return to previous menu.

Returns SDEMO to the Main Functions window (pressing [esc] does the same).

2. Access NIC registers and display buffer memory.

This menu (figure 2.0) allows read/write/display access to NIC registers and buffer memory.

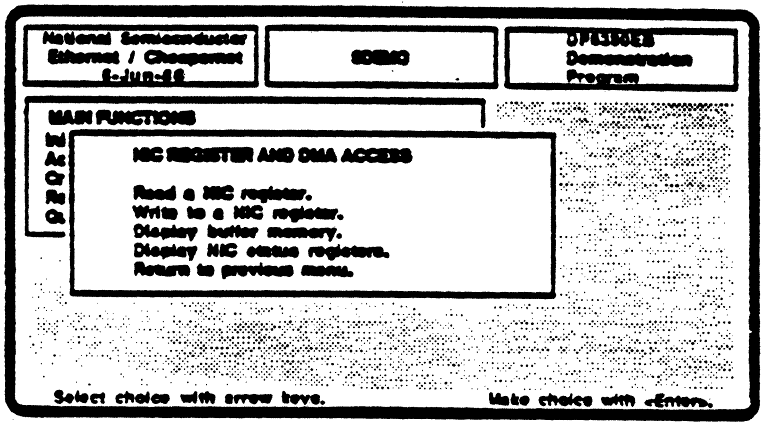


Figure 2.0

2.1. Read a NIC register.

This option allows you to display the contents of any register without altering the contents. The window prompts for a port number in hex. Enter the desired address and press [return]. The contents of that port will be displayed within the window. Allowed addresses are 0000h → FFFFh.

2.2. Write to a NIC register

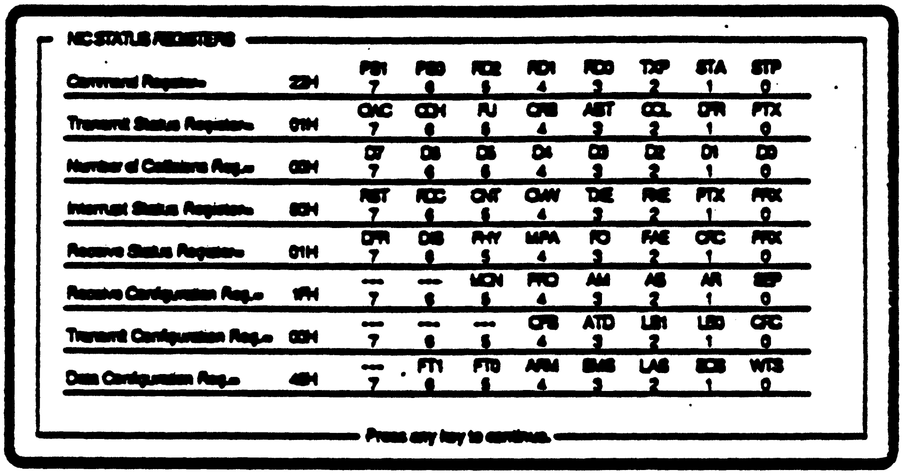
Using this option you can write a byte of data into any register. The window first prompts for the port number in hex and then for the data byte (in hex) to be written.

2.3. Display buffer memory

This selection is used to display a portion of the buffer memory. You will be prompted to first enter in hex the starting address and next for how many bytes you would like to display. Allowed addresses are 0000h → FFFFh. NIA ROM starts at 0000h and ends at 0020h. NIA RAM starts at 2000h and ends at 4000h. SDEMO will indicate that a DMA is in process and then display the specified buffer memory space in hex. Pressing any key during display will terminate the display.

2.4. Display NIC status registers.

This option opens a window (figure 2.4) which displays the contents of the NIC's status registers. Each of the eight rows contain the register's name followed by it's value in hex. Also displayed are the register's bit names and numbers, with the numbers of the set bits in each register highlighted by inverse video. Press any key to continue.



NIC STATUS REGISTERS

Command Register	22H	PS1	PS0	RD2	RD1	RD0	TXP	STA	STP
		7	6	5	4	3	2	1	0
Transmit Status Register	01H	OWC	CDH	FU	CRS	ABT	COL	DFR	PTX
		7	6	5	4	3	2	1	0
Number of Collisions Reg.	00H	D7	D6	D5	D4	D3	D2	D1	D0
		7	6	5	4	3	2	1	0
Interrupt Status Register	80H	RST	RDC	CNT	OVW	TXE	RXE	PTX	PRX
		7	6	5	4	3	2	1	0
Receive Status Register	01H	DFR	DIS	PHY	MPA	FO	FAE	CRC	PRX
		7	6	5	4	3	2	1	0
Receive Configuration Reg.	1FH	---	---	MON	PRO	AM	AB	AR	SEP
		7	6	5	4	3	2	1	0
Transmit Configuration Reg.	00H	---	---	---	OFST	ATD	LB1	LB0	CRC
		7	6	5	4	3	2	1	0
Data Configuration Reg.	48H	---	FT1	FT0	ARM	BMS	LAS	BOS	WTS
		7	6	5	4	3	2	1	0

Press any key to continue.

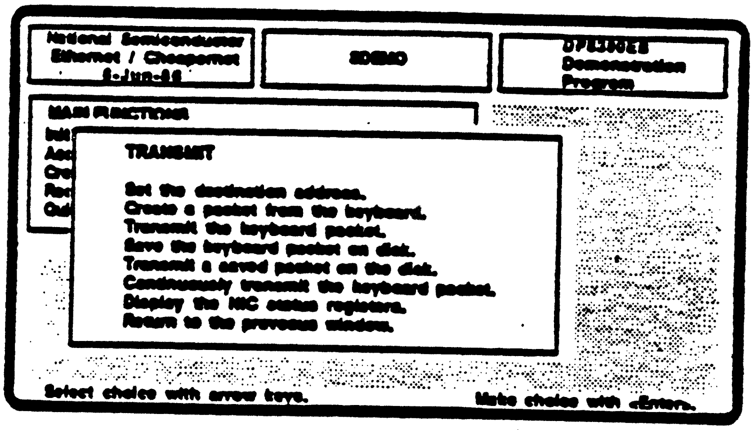
Figure 2.4

2.5. Return to previous menu

Returns SDEMO to the Main Functions window (pressing [esc] does the same).

3. Create and transmit packets.

This menu (figure 3.0) accesses SDEMO's packet management and transmission routines.



TRANSMIT

Set the destination address.
Create a packet from the keyboard.
Transmit the keyboard packet.
Save the keyboard packet on disk.
Transmit a saved packet on the disk.
Continuously transmit the keyboard packet.
Display the NIC status registers.
Return to the previous window.

Select choice with arrow keys. Make choice with <Enter>.

Figure 3.0

3.1. Set the destination address.

This selection prompts for a twelve digit destination address (in hex) for the packet.

3.2. Create a packet from the keyboard.

Use this option to create a packet. The packet must not exceed 4096 characters. The [return] key enters the packet into memory and terminates the window. The previous keyboard packet will be overwritten in the process.

3.3. Transmit the keyboard packet.

Selecting this option transmits a single keyboard packet onto the network.

3.4. Save the keyboard packet on disk.

This option allows you to save the keyboard packet onto the disk for future use. Enter a valid DOS filename and press [return].

3.5. Transmit a saved packet on the disk.

Select this option to transmit a packet which has been previously stored on a disk. Enter the filename and press [return]. At this point pressing any key will transmit the packet and return to the transmit window.

3.6. Continuously transmit the keyboard packet.

In this mode the keyboard packet is continuously transmitted. The window prompts for a delay factor between transmissions. The delay is linear (a delay of eight is twice as long as four) and may be between zero and 32000 inclusive. Each transmitted packet is indicated by a dot printed on the screen. Pressing any key terminates transmission and returns to the previous window.

3.7. Display the NIC status registers.

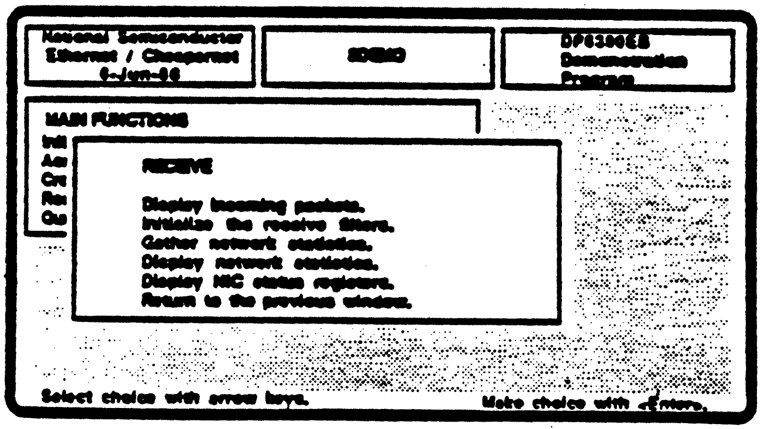
This selection is equivalent to window 2.4 described previously.

3.8. Return to the previous window.

Returns SDEMO to the Main Functions window (pressing [esc] does the same).

4. Receive and display incoming packets.

This menu (figure 4.0) accesses SDEMO's packet display and statistics gathering routines.



National Semiconductor
Ethernet / Cheapernet
6-Jun-86

SDEMO

DP8390EB
Demonstration
Program

MAIN FUNCTIONS

RECEIVE

Display incoming packets.
Initialize the receive filters.
Gather network statistics.
Display network statistics.
Display NIC status registers.
Return to the previous window.

Select choice with arrow keys.

Make choice with <Enter>.

Figure 4.0

4.1. Display incoming packets.

Selecting this option displays incoming packets which pass through the receive filters. The left column displays the packet in hex while the right column displays it's ASCII equivalent. Pressing any key terminates the display and returns to the previous window.

4.2. Initialize the receive filters.

This selection is equivalent to window 1.0 described previously.

4.3. Gather network statistics.

This option allows for monitoring network traffic. Each accepted packet is indicated by a dot printed on the screen. The number of accepted packets while in this mode is stored in the number of packets counter which rolls over at 32000. Errored packets are also tallyed and these counters roll over at 192. Pressing any key ends statistic gathering and returns to the receive window.

4.4. Display network statistics.

This selection opens a window which displays the network statistics. It indicates how many packets were received and how many were lost due to frame alignment errors, CRC errors and buffer overflow. Press any key to exit this window.

4.5. Display NIC status registers.

This selection is equivalent to window 2.4 described previously.

4.6. Return to the previous window.

Returns SDEMO to the Main Functions window (pressing [esc] does the same).

5. Quit SDEMO and return to DOS.

Terminate the current SDEMO session and reenter the DOS environment (pressing [esc] does the same).

National
Semiconductor
Corporation

Network Load Simulator

NLS.EXE Preliminary
July 1986

OVERVIEW

The Network Load Simulator program (NLS) is a useful Local Area Network (LAN) development tool for evaluating and testing driver level software. Packets are transmitted in bursts with menu selected packet sizes, interpacket delays, burst sizes and interburst delays as depicted in figure 1. All parameters can be set to fixed values or may be set to vary linearly or bimodally within a selected range. By using varying delays and packet sizes, traffic on an active LAN can be emulated, and driver software can be tested and evaluated in a typical environment. Packets sent out contain a MOD 15 counting pattern which can be conveniently verified by the receiving nodes (NES supports checking of this pattern).

The NIA is an IBM PC-compatible LAN demonstration board designed to evaluate National's IEEE 802.3-compatible DP8390 network chip set. The DP8390 chip set is one of the latest of National's Advanced Peripheral Processing Solutions (APPS:) family of VLSI circuits for microprocessor peripheral interface applications. The board plugs into any PC-compatible machine and incorporates all of the components required to provide a LAN interface to Ethernet or Cheapernet networks. An optional daughter card allows LAN interface with low cost STARLAN networks.

FEATURES

- GENERATES NETWORK TRAFFIC
- FLEXIBLE PACKET GENERATOR
- A DRIVER DEVELOPMENT TOOL
- MENU DRIVEN

REQUIREMENTS

NLS requires an NIA equipped PC. **The NIA board jumpers must be configured to use I/O space 2E0 (serial port 2)** as described in the I/O Base Address section of the Network Interface Adapter Hardware Reference Manual.

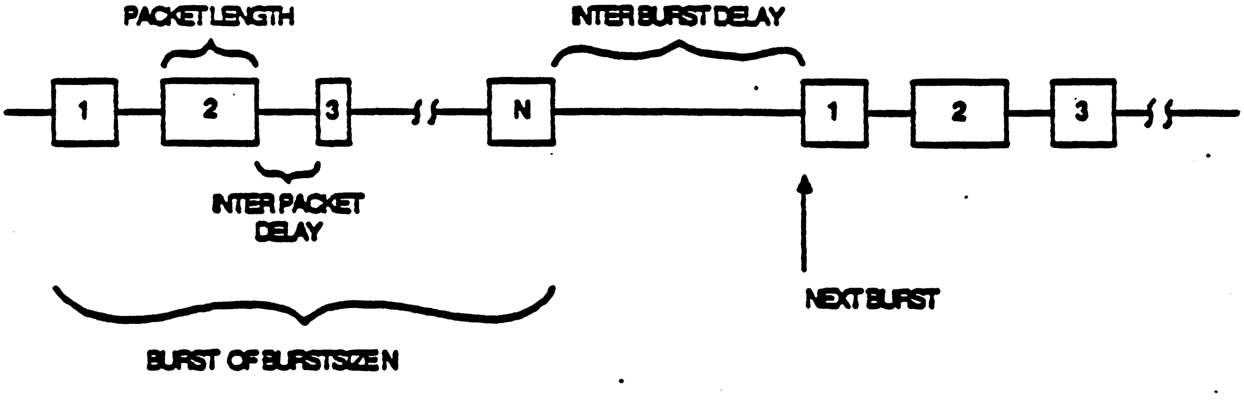
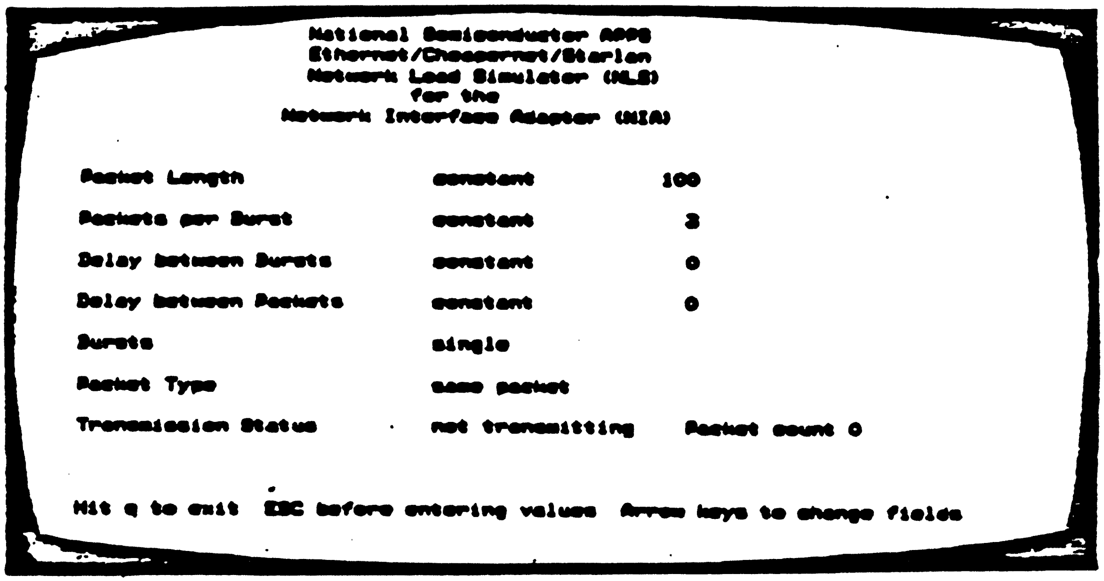


Figure 1

LOADING AND RUNNING

Boot the PC with DOS 2.0 or greater. Insert the DEMO disk into the active disk drive. Invoke NLS from the keyboard by typing NLS [return]. The terminal will display the NLS menu as depicted below in figure 2.



```
                National Semiconductor APPS
               Ethernet/Cheapernet/Starlan
               Network Load Simulator (NLS)
                         for the
             Network Interface Adapter (NIA)

Packet Length               constant            100

Packets per Burst           constant              3

Delay between Bursts        constant              0

Delay between Packets       constant              0

Bursts                      single

Packet Type                 same packet

Transmission Status         not transmitting    Packet count 0


Hit q to exit  ESC before entering values  Arrow keys to change fields
```

Figure 2

Moving the Cursor

The items on the menu are accessed using the cursor control keys of the PC's numeric keypad. Pressing the [2↓] key moves the cursor down the menu, [8↑] moves it up, [4←] to the left and [6→] to the right. In the vertical direction the menu wraps around itself. Note: If the cursor control keys are not performing these functions press the [NumLock] key.

Special Keys

There are five special keys used in NLS. These are [t], [r], [SpaceBar], [esc] and [q] and their functions are as follows:

[t]: This key brings the cursor from anywhere on the menu to the last row and enters the transmit mode.

[r]: Reset packet count. The packet count is the number of packets sent out on the network since the count was last reset.

[SpaceBar]:
The [SpaceBar] key is used to toggle the possible selections in the second column.

[esc] The escape key must be depressed before entering values (see Selecting Value on the following page).

[q]: Use this option to exit NLS and re-enter the DOS environment.

Note: Pressing any key while continuously transmitting terminates transmission.

Distribution of Packet Lengths, Packets per Burst and Delays

The distribution of packet lengths, packets per burst, delays between bursts and delays between packets have three possibilities: constant, linear and bimodal. To select between these three possibilities simply move the cursor to the second column and the desired row and press the [SpaceBar] as mentioned previously.

Constant: Displayed value used constantly.

Linear: A psuedo-random linear distribution between the two values displayed.

Bimodal: A psuedo-random distribution with 65% of first value and 35% of the second value.

Note: Delays increase with the square of the number, with zero being the smallest delay.

Selecting Values

Values are entered by moving the cursor to the appropriate position on the menu and then pressing the escape [esc] key. The desired value may now be typed in and entered by pressing [return]. For a linear distribution the left number must be less than the right number.

Bursts may be selected as occurring singularly or continuously.

There are two options for packet type; same packet or different packets. NLS initially loads the transmit buffer with a series of 256 byte pages starting at page 20h. When the Same Packet option is selected the NIC transmits the packet beginning at page 20h and consists of the pattern shown in figure 3. Different packets mode transmits the packets in locations 21h through 21+f (the beginning of 21h is shown below).

```
2000   88 77 66 55 44 33 00 01 02 03 04 05 06 07 08 09
2010   0A 0B 0C 0D 0E 00 01 02 03 04 05 06 07 08 09 0A
2020   0B 0C 0D 0E 00 01 02 03 04 05 06 07 08 09 0A 0B
2030   0C 0D 0E 00 01 02 03 04 05 06 07 08 09 0A 0B 0C
2040   0D 0E 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D
2050   0E 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E
2060   00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 00
2070   01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 00 01
2080   02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 00 01 02
2090   03 04 05 06 07 08 09 0A 0B 0C 0D 0E 00 01 02 03
20A0   04 05 06 07 08 09 0A 0B 0C 0D 0E 00 01 02 03 04
20B0   05 06 07 08 09 0A 0B 0C 0D 0E 00 01 02 03 04 05
20C0   06 07 08 09 0A 0B 0C 0D 0E 00 01 02 03 04 05 06
20D0   07 08 09 0A 0B 0C 0D 0E 00 01 02 03 04 05 06 07
20E0   08 09 0A 0B 0C 0D 0E 00 01 02 03 04 05 06 07 08
20F0   09 0A 0B 0C 0D 0E 00 01 02 03 04 05 06 07 08 09
2100   0A 0B 0C 0D 0E 00 01 02 03 04 05 06 07 08 09 0A  ←Begin new page
2110   0B 0C 0D 0E 00 01 02 03 04 05 06 07 08 09 0A 0B   (page 21h)
2120   0C 0D 0E 00 01 02 03 04 05 06 07 08 09 0A 0B 0C
2130   0D 0E 00 01 02 03 04 05 06 07 ....etcetera....
```

Figure 3

National
Semiconductor
Corporation

Network Evaluation Software
NES.EXE Preliminary
July 1988

OVERVIEW

NES is an assembly language program used to evaluated the NIC in a real-life network environment. NES is composed of .two object modules, nes.obj and isrb3a.obj. The main program module is nes.obj which simulates an operating system shell. The isrb3a.obj module contains the interrupt service routine which handles low-level receive and transmit processing. NES also serves as an example of a driver for upper-level network software.

FEATURES

- EVALUATES DP8390 IN A NETWORK ENVIRONMENT
- TRANSMIT AND RECEIVE ERROR DETECTION
- DISPLAYS NETWORK STATISTICS

REQUIREMENTS

NES requires the NIA to be configured using Serial Port 2, DMA Channel 1 and interrupt request 3 (the factory installed configuration). Refer to the switch settings section of the NIA Hardware reference Manual.

If needed, the address base, interrupt request number and the DMA request number can be changed from their default values of:

address base = 2E0h interrupt request number = IREQ3 DMA request number = DRQ1

In the equate section at the beginning of each object module there are "comment lines" provided to facilitate changing to the other NIA board hardware options (address base = 260h, IREQ5 andDRQ3). The comment lines are as follows:

```
Option 1
        COMMAND            equ   2E0h
      ;   COMMAND          equ   260h      ;"comment line" → address base 260h

 Option 2
        IRQ3               equ   0F7h
        CODE_IRQ3          equ   2Ch
      ; IRQ5               equ   0DFh
      ; CODE_IRQ5          equ   44h

Option 3
        ENB_DRQ1           equ   1
        DISABLE_DRQ1       equ   5
      ; ENB_DRQ3           equ   3
      ; DISABLE_DRQ3       equ   7
```

To change the NIA hardware options exchange the commented lines for the commented lines. For example, to change the address base to 260h change option 1 to:

```
        COMMAND            equ   260h
      ; COMMAND            equ   2E0h
```

LOADING AND RUNNING

To execute this program, type "nes [options]" on the DOS command line. The options allow you to reconfigure the NIC with different network parameters. The options are as follows: C, T, I, and H, and may be selected in upper or lower case. Examples of invoking the program are shown below.

```
A>nes       no options selected
A>nes c     the "c" option selected
A>nes c t   the "c" and "t" option select,
            the  order does not matter
```

PROGRAM OPTIONS

The "C" option allows you to check all incoming packets with a chosen destination address (first 6 bytes) and the following data pattern - 0 1 2 3 4 5 6 7 8 9 a b c d e 0 1 2 etc. (the pattern generated by NLS). The chosen address is prompted by the program when the "C" option is selected. If an error occurs, it will be indicated on the screen.

The "I" options allows you to change the NIC configuration registers and interrupt mask register. When this option is selected, you will be prompted to change the Data Configuration Register (DCR), the Transmit Configuration Register (TCR), the Receive Configuration Register (RCR), and Interrupt Mask Register (IMR). The default values are indicated in brackets ([xx]). To change these registers, enter the appropriate hex value; otherwise hit <return> to leave the register unchanged. Note that if this option is not selected, the NIC is configured with the default values shown in the brackets.

The "H" option displays all good packets received.

The "T" option allows you to continuously transmit packets with a destination address prompted by the program.

These options may be selected singly or together and may be in any order.

SCREEN DISPLAY

When the program is executing, various messages and characters are displayed to indicate good and bad reception and transmissions. The meanings of the characters are as follows:

•	good packet received.
n	a packet receive interrupt was set but there was nothing in the Receive Buffer Ring. Action taken: exit the interrupt service routine.
b	packet received but with bad receive status (anything other than 01h or 21h) in NIC header. Action taken: ignore the packet and move on to the next one, if any.
d	packet received but the DMA did not complete. Action taken: reattempt to DMA the packet again.
r	packet received but the Next Page Pointer in the NIC header was outside the limits of the Receive Buffer Ring. Action taken: reset the BOUNDARY and CURRENT registers to their initial conditions.
l	packet received but too long (>1500). Action taken: ignore packet and move on to the next packet, if any.
t	good packet transmitted.

NOTE: There are other messages to indicate transmission errors such as Excessive Collisions, Carrier Sense Lost, and FIFO Underrun.

X exiting from the interrupt service routine.

Messages:

- *** Bad address in packet *** [address of packet] This message occurs if there is a mismatch between the address of the incoming packet and the expected address that was entered with the "C" option.
- *** Bad data in packet *** [location where error occurred] This message occurs if the incoming packet deviated for the pattern 0,1,2,...,E,0,1,2,...,etc. and the "C" option was selected.

The program has two keyboard commands, ^E and any other key. When ^E is typed the program displays network statistics (as shown below) then exits and returns to DOS. The program does not exit until the interrupt service routine has exited (ie. after an "X" has been displayed). Hitting any other key will transmit a packet.

```
      NETWORK STATISTICS COUNT (hex)

Number of good transmissions:   0000 0000
Number of bad transmissions:         0000
Number of good receptions:      0000 0000
Number of bad receptions:            0000
Number of Next Page Pointer errors:  0000
```

National
Semiconductor
Corporation

Writing Drivers for the DP8390

Preliminary
July 1986

INTRODUCTION

To reduce the complexity of networking software, it is organized as a series of layers. The number of layers, the name of each layer, and the function of each layer varies from network to network. However, in all networks, the purpose of each layer is to offer certain services to its adjacent layers and to shield those layers from how the offered services are actually implemented. Figure 1 illustrates a distributed software model consisting of (1) an applications layer, (2) an upper level communications layer, (3) a driver layer, and (4) the DP8390.

The two upper layers form the bulk of the software model and are beyond the scope of this document. This document is intended to provide the information necessary to write a network communication driver to provide certain services for upper layer software.

CUSTOM DRIVER PROCEDURES

The drivers consist of several custom-written procedures which will provide the following services:

- Hardware initialization
- Packet transmission
- Packet reception
- Detecting error conditions

The drivers should be able to run in an interrupt-driven environment which responds to the various interrupts produced by the DP8390. To achieve the highest performance the drivers should have as little overhead as possible.

INITIALIZATION

The initialization procedure configures the DP8390 to conform to the present network. This involves initializing the proper configuration and address registers of the DP8390. A summary of the network parameters involved are shown below.

- the size of the data bus (8 or 16 bits)
- the physical and multicast addresses
- the types of interrupts used
- the size of the Receive Buffer Ring
- the size of the transmit buffer
- the FIFO threshold

An example of an initialization routine for a typical network system is provided at the end of this document (DriverInitialize).

PACKET TRANSMISSION

The transmit driver is responsible for making a "best effort" delivery to the destination node. By "best effort" we mean that the drivers ensure that a packet will be delivered with a high probability of success. This is not to say that the driver must make a 100% delivery; this is the responsibility of the upper layer software. Since the DP8390 implements a binary backoff algorithm which retransmits up to 15 times in the event of collisions, some "best effort" delivery is already built into the hardware.

The transmit drivers are generally partitioned into two parts. The first part (DriverSend) initiates a transmission whenever the upper level software desires to send a packet.. If the transmitter is not ready, however, the supplied packet should be queued in a transmit-pending buffer. After transmit initiation or queueing, DriverSend returns.

DriverSend operates in conjunction with an interrupt service routine (DriverISR). After completing the transmission, the DP8390 interrupts the CPU to signal the end of the transmission and to indicate status information. Since the transmit drivers can not assure delivery, they must inform the upper level software of successful or errored transmissions. The DP8390 reports this status information in its TRANSMIT STATUS REGISTER.

The Transmit Queue

In many instances, a queue for transmitted packets is not required. A queue will be required if the rate at which packets can be generated by the upper level communications routine is higher than the rate at which they can be delivered on the network.

The recommended method for implementing a queue is a linked list as shown in figure 2. The queue consists of two pointers, head_ptr and tail_ptr, which respectively point to the next buffer space available and next buffer to be read. At the beginning of each buffer containing a packet, a pointer field points to the next packet in the queue, and the last buffer points to head_ptr. Each time a packet is added or removed, either the head_ptr or tail_ptr is incremented. The queue should be circular where pointers "wrap around" when the boundaries of the queue have been reached.

The most efficient manner to remove packets from the transmit- pending queue is to use DriverSend to initiate transmission of the very first packet in the queue; then upon completion, use the DriverISR to transmit the remaining packets. Using this method, the DriverISR examines the queue, transmits the next available packet, then exits. The DriverISR transmits the next packet after the DP8390 issues the next transmit interrupt.

RECEIVE PROCESSING

The responsibility of the receive drivers is to buffer incoming packets from the network, and then to transfer them to the host. In most systems, the receive drivers will be operating with two different memory banks. One is the local memory which the DP8390 uses to buffer packets from the network; the other is the host memory. Depending upon which architecture is used, local memory can be dual ported or shared by the host CPU. In shared memory configurations, the host CPU intervenes directly with local memory. In dual ported systems, local memory is shielded from the host and data is transferred through a DMA channel.

The DP8390 minimizes the task of the receive driver with its unique buffer management system which buffers packets into local memory (Receive Buffer Ring) automatically. All the receive driver must do is remove packets from the Receive Buffer Ring. In the shared memory systems, the host removes packets by directly accessing local memory. This system, however, can lead to some tricky arbitration problems. In the dual ported system which provides a cleaner interface, the receive driver issues a DMA command to the DP8390 to transfer data from local to host memory. Again, the DP8390 simplifies packet reception with its "send packet" command. Using this command, the Receive Buffer Ring pointers (BOUNDARY and CURRENT registers) are maintained by the DP8390. To remove all packets from local memory, the receive driver simply issues the "send packet" command until the BOUNDARY and CURRENT registers are equal (i.e. the Receive Buffer Ring is empty).

Obviously, because of the asynchronous nature of reception, the receive drivers must reside within an interrupt service routine (DriverISR). Typically, packet reception is given the highest priority interrupt since prolonging packet removal may overflow the Receive Buffer Ring. If several packets in the ring have been queued, all packets should be removed in one process (i.e. a software loop which empties the Receive Buffer Ring). In heavy traffic conditions, local memory can fill up very quickly and it is important that the Receive Buffer Ring be large enough to handle these situations.

To find out how many packets are lost on the network and how many packets are lost due to Receive Buffer Ring overflows, the DP8390 has three statistical registers to monitor the network: FRAME ALIGNMENT ERROR tally, CRC ERROR tally, and FRAMES LOST tally. These registers are useful in determining the size of the Receive Buffer Ring and how many packets are lost due to network related errors (CRC errors and/or frame alignment errors).

EXAMPLE DRIVERS

The following transmit and receive drivers are written in 8086 assembly for high performance and low overhead. The transmit driver is partitioned into two parts, DriverSend and DriverISR, while the receive driver lies entirely within DriverISR. This section first gives an overview of DriverISR, followed by a description of the drivers and how they interact with DriverISR.

Interrupt Service Routine (DriverISR)

DriverISR is only concerned with interrupts originating from receptions, transmissions, and errored transmissions. Errored receptions are ignored since these are usually collision fragments. DriverISR (figure 3) consists of (1) a packet transmitted routine and (2) a packet received routine. Either receive or transmit interrupts may use both routines since receptions and transmissions can occur within the ISR. The basic functions of the routines are as follows:

(1) Packet Transmitted Routine: checks the status of all transmissions and transmits the next packets in the transmit-pending queue.

(2) Packet Received Routine: removes all packets in the receive buffer ring by using the "send packet" command of the DP8390.

Transmit Drivers

As mentioned before, the transmit drivers consist of two parts. The first part, DriverSend (figure 4), initiates transmission when called by the upper layer software. DriverSend checks if the DP8390 is ready to transmit by reading the COMMAND register (CR = 22h). If ready, the DriverSend DMAs a packet from host to local memory, issues the transmit command, then returns. Otherwise, DriverSend queues the packet in the transmit-pending queue, then returns.

After a transmission is completed, DriverISR services the interrupt from the DP8390 and (1) reports status information by reading the TRANSMIT STATUS register and (2) transmits the next packet in the transmit-pending queue, if any. For a transmit interrupt, DriverISR executes the following steps:

(1) Reset PTX bit in INTERRUPT STATUS register.

(2) Check for good transmission by reading the TRANSMIT STATUS Register.

(3) If there are more packets in the transmit-pending queue, transmit the next packet; otherwise go to 4).

(4) Read INTERRUPT STATUS register for any pending interrupts

Receiver Drivers

Since the receive driver must be interrupt driven, it resides completely within the DriverISR. When the receive interrupt occurs, one or more packets may be buffered into the receive buffer ring (RBR) by the DP8390. The DriverISR removes packets from the ring and then passes them up to the host. Using the "send packet" command, packets are removed until the RBR is empty, that is, when CURRENT and BOUNDARY registers are equal. The sequence of the receive packet routine is shown below.

(1) Reset the PRX bit in the INTERRUPT STATUS register.

(2) Remove the next packet in the receive buffer using the "send packet" command.

(3) Check to see if the receive buffer ring is empty: BOUNDARY register = CURRENT PAGE register

(4) If the RBR is not empty, goto 1; otherwise read INTERRUPT STATUS register for any more pending interrupts.

Upper Layer Software Interface

The drivers described exemplify the basic method for using the DP8390 in a network environment. The upper layer interface, however, is still lacking. Depending upon which software you use, the protocols for communicating to the upper layers can vary greatly. The source listings at the end of the document have patches for this upper layer interface.

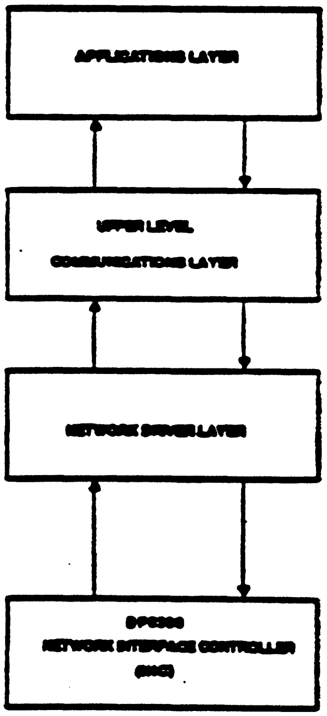


FIGURE 1
BASIC DISTRIBUTED SOFTWARE MODEL

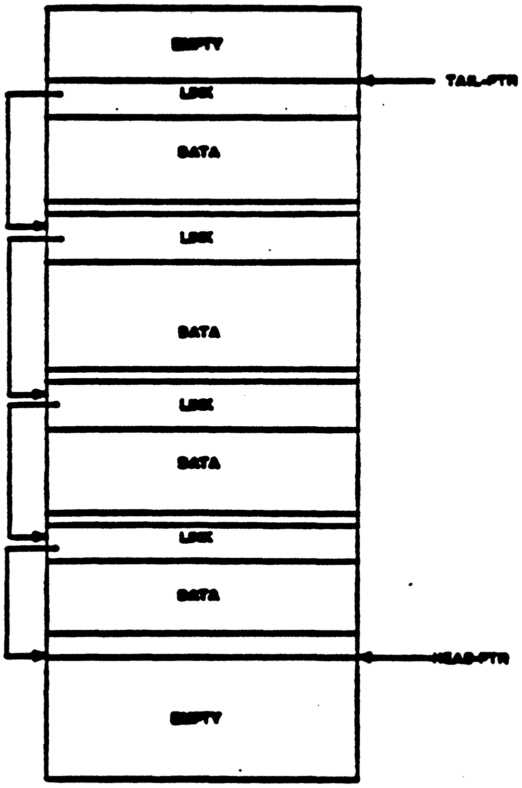


FIGURE 2
TRANSMIT-PENDING QUEUE

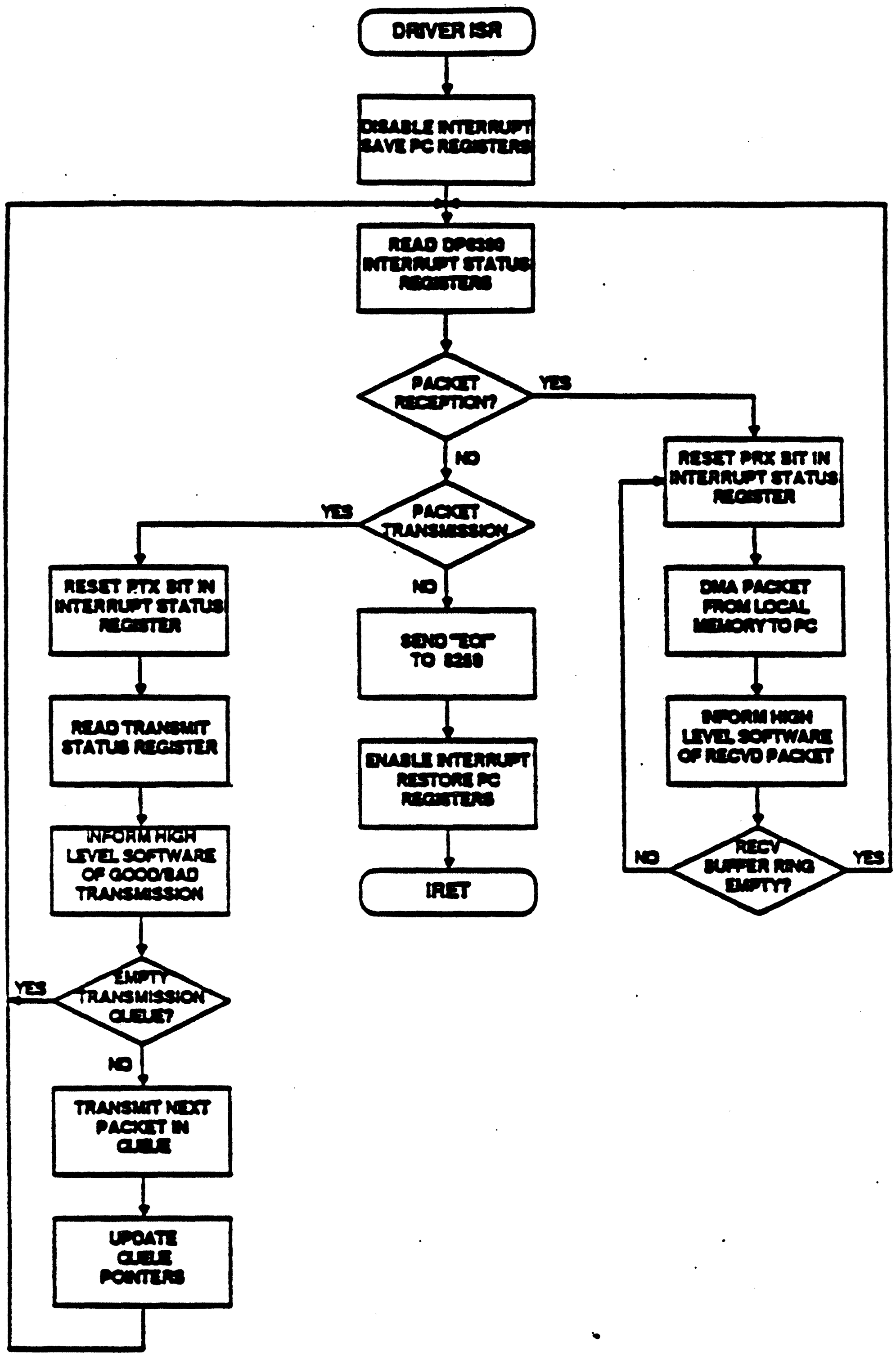

DRIVER ISR
DISABLE INTERRUPT SAVE PC REGISTERS
READ DP8390 INTERRUPT STATUS REGISTERS
PACKET RECEPTION?
YES
NO
PACKET TRANSMISSION
YES
NO
RESET PTX BIT IN INTERRUPT STATUS REGISTER
READ TRANSMIT STATUS REGISTER
INFORM HIGH LEVEL SOFTWARE OF GOOD/BAD TRANSMISSION
EMPTY TRANSMISSION QUEUE?
YES
NO
TRANSMIT NEXT PACKET IN QUEUE
UPDATE QUEUE POINTERS
SEND "EOI" TO 8259
ENABLE INTERRUPT RESTORE PC REGISTERS
IRET
RESET PRX BIT IN INTERRUPT STATUS REGISTER
DMA PACKET FROM LOCAL MEMORY TO PC
INFORM HIGH LEVEL SOFTWARE OF RECVD PACKET
RECV BUFFER RING EMPTY?
NO
YES

FIGURE 3
INTERRUPT SERVICE ROUTINE
(ISR)

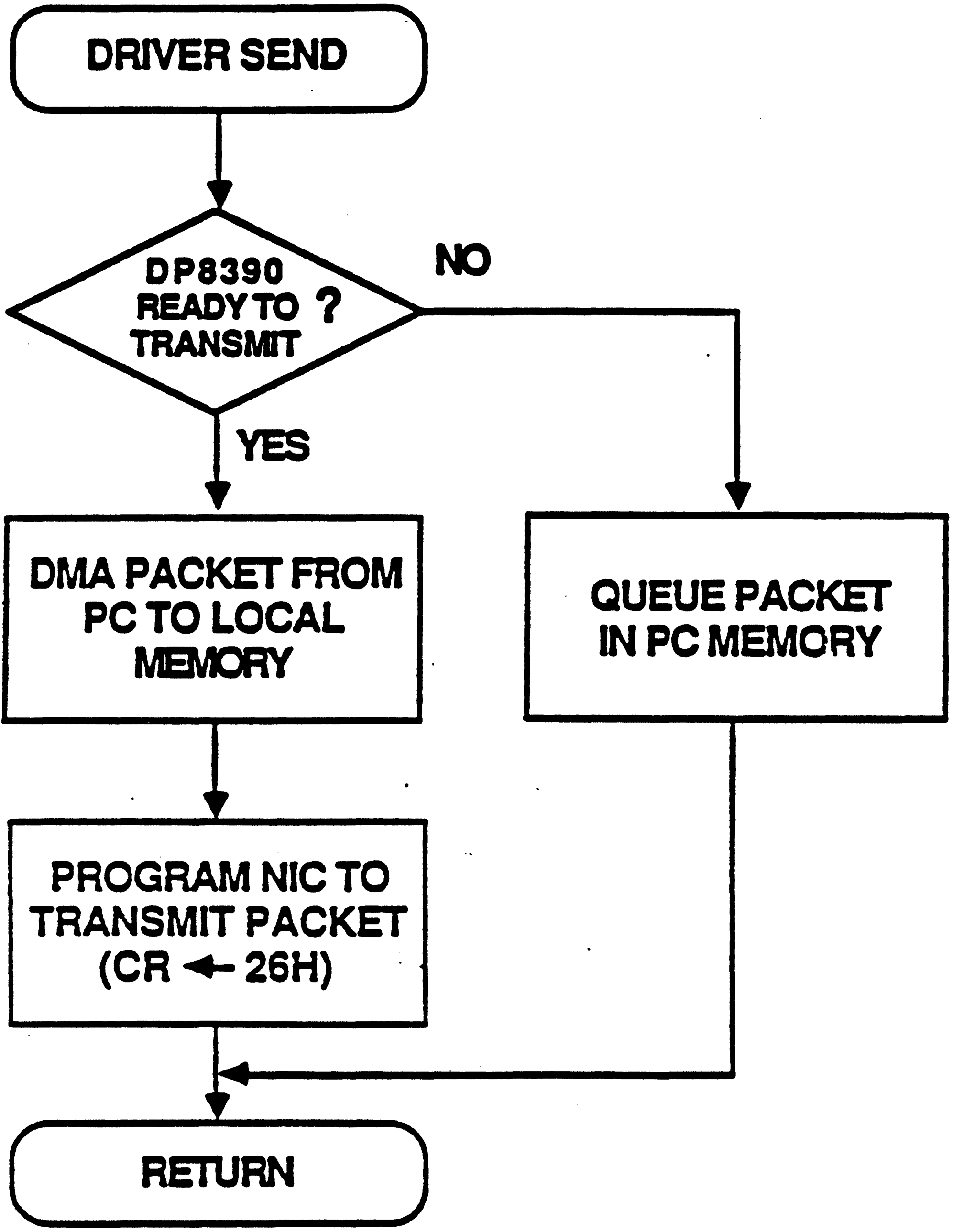

DRIVER SEND
DP8390 READY TO TRANSMIT ?
NO
YES
DMA PACKET FROM PC TO LOCAL MEMORY
QUEUE PACKET IN PC MEMORY
PROGRAM NIC TO TRANSMIT PACKET (CR ← 26H)
RETURN

FIGURE 4
DRIVER SEND ROUTINE

```
;•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
;                          DriverInitialize
;
;   Initializes the NIC for a typical network system.
;   Receive Buffer Ring = 2600h to 4000h
;   Transmit Buffer     = 2000h to 2600h
;
;        Entry: none
;
;•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••

;•••••••••••••••••••• Equates for NIC registers ••••••••••••••••••••

COMMAND                 equ     2e0h
PAGESTART               equ     COMMAND+1
PAGESTOP                equ     COMMAND+2
BOUNDARY                equ     COMMAND+3
TRANSMITSTATUS          equ     COMMAND+4
TRANSMITPAGE            equ     COMMAND+4
TRANSMITBYTECOUNT0      equ     COMMAND+5
NCR                     equ     COMMAND+5
TRANSMITBYTECOUNT1      equ     COMMAND+6
INTERRUPTSTATUS         equ     COMMAND+7
CURRENT                 equ     COMMAND+7           ;in page 1
REMOTESTARTADDRESS0     equ     COMMAND+8
CRDA0                   equ     COMMAND+8
REMOTESTARTADDRESS1     equ     COMMAND+9
CRDA1                   equ     COMMAND+9
REMOTEBYTECOUNT0        equ     COMMAND+0ah
REMOTEBYTECOUNT1        equ     COMMAND+0bh
RECEIVESTATUS           equ     COMMAND+0ch
RECEIVECONFIGURATION    equ     COMMAND+0ch
TRANSMITCONFIGURATION   equ     COMMAND+0dh
FAE_TALLY               equ     COMMAND+0dh
DATACONFIGURATION       equ     COMMAND+0eh
CRC_TALLY               equ     COMMAND+0eh
INTERRUPTMASK           equ     COMMAND+0fh
MISS_PKT_TALLY          equ     COMMAND+0fh

PSTART                  equ     26h
PSTOP                   equ     40h

CGroup  group  Code
Code    segment para public 'Code'
        assume  cs:CGroup, ds:CGroup, es:nothing, ss:nothing

rcr     db      0                   ;value for Recv config. reg
tcr     db      0                   ;value for trans. config. reg
dcr     db      58h                 ;value for data config. reg
imr     db      0bh                 ;value for intr. mask reg
;•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
DriverInitialize proc near
        public DriverInitialize

        mov     al,21h                  ;stop mode
        mov     dx,COMMAND
        out     dx,al
        mov     al,dcr
        mov     dx,DATACONFIGURATION    ;data configuration register
        out     dx,al
        mov     al,20h
        mov     dx,TRANSMITPAGE         ;transmit page start
        out     dx,al
        mov     al,tcr
        mov     dx,TRANSMITCONFIGURATION ;transmit configuration register
        out     dx,al
        mov     al,rcr
        mov     dx,RECEIVECONFIGURATION  ;receive configuration register
        out     dx,al
        mov     al,26h
        mov     dx,PAGESTART             ;page start
        out     dx,al
        mov     dx,BOUNDARY              ;boundary register
        out     dx,al
        mov     al,40h
        mov     dx,PAGESTOP              ;page stop
        out     dx,al
        mov     dx,REMOTEBYTECOUNT0
        mov     al,0fh
        out     dx,al                    ;low remote byte count
        mov     dx,REMOTEBYTECOUNT1
        mov     al,00
        out     dx,al                    ;high remote byte count
        mov     al,0ffh
        mov     dx,INTERRUPTSTATUS       ;interrupt status register
        out     dx,al
        mov     al,imr
        mov     dx,INTERRUPTMASK         ;interrupt mask register
        out     dx,al
        mov     al,61h                   ;go to page 1 registers
        mov     dx,COMMAND
        out     dx,al
        mov     al,26h
        mov     dx,CURRENT               ;current page register
        out     dx,al
        mov     al,22h                   ;back to page 0, start mode
        mov     dx,COMMAND
        out     dx,al

        ret
DriverInitialize endp

Code    ends
        end
```

```
;••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
;                        DriverSend
;
; Either transmits the packet passed to it or queues up the packet if
; the transmitter is busy (COMMAND register = 26h).  Routine is called
; from upper layer software.
;
;      Entry:   es:si -> packet to be transmitted
;
;••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••


;•••••••••••••••••••• Equates for NIC registers ••••••••••••••••••••


COMMAND                 equ     2e0h
PAGESTART               equ     COMMAND+1
PAGESTOP                equ     COMMAND+2
BOUNDARY                equ     COMMAND+3
TRANSMITSTATUS          equ     COMMAND+4
TRANSMITPAGE            equ     COMMAND+4
TRANSMITBYTECOUNT0      equ     COMMAND+5
NCR                     equ     COMMAND+5
TRANSMITBYTECOUNT1      equ     COMMAND+6
INTERRUPTSTATUS         equ     COMMAND+7
CURRENT                 equ     COMMAND+7               ;in page 1
REMOTESTARTADDRESS0     equ     COMMAND+8
CRDA0                   equ     COMMAND+8
REMOTESTARTADDRESS1     equ     COMMAND+9
CRDA1                   equ     COMMAND+9
REMOTEBYTECOUNT0        equ     COMMAND+0ah
REMOTEBYTECOUNT1        equ     COMMAND+0bh
RECEIVESTATUS           equ     COMMAND+0ch
RECEIVECONFIGURATION    equ     COMMAND+0ch
TRANSMITCONFIGURATION   equ     COMMAND+0dh
FAE_TALLY               equ     COMMAND+0dh
DATACONFIGURATION       equ     COMMAND+0eh
CRC_TALLY               equ     COMMAND+0eh
INTERRUPTMASK           equ     COMMAND+0fh
MISS_PKT_TALLY          equ     COMMAND+0fh

PSTART                  equ     26h
PSTOP                   equ     40h

CGroup  group   Code
Code    segment para public 'Code'
        assume  cs:CGroup, ds:CGroup, es:nothing, ss:nothing

;••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••

DriverSend      proc    near
        public      DriverSend

        cli                             ;disable interrupts
        mov     dx,COMMAND
        in      al,dx                   ;read NIC command register
        cmp     al,26h                  ;transmitting?
        je      QueueIt                 ;yes, queue up the packet
```

```
;    ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;    Before we can DMA a packet to local memory, we need the address
;    and byte count of the packet passed down from the upper layer
;    software; two routine are used for DMAing.
;       (1) Set_PC_DMA_address sets up the DMA address of the 8237
;           (requires that bx = address of packet).
;       (2) dma2nic sets of the 8327 and the NIC to perform a
;           DMA operation (routine requires that
;           ax = byte count of packet).
;    ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

        mov     bx,si                   ;si points to the packet passed down
                                        ;  from the upper layer software
        mov     ax,byte_count           ;need byte count of packet
                                        ;  also should have been passed down
                                        ;  from upper layer software
        call    PageWrapCheck           ;check to see if DMA will cross over
                                        ;  a 64K byte boundary.  If it does,
                                        ;  we must break the DMA into 2 parts.
        cmp     cx,0                    ;return code from PageWrapCheck
                                        ;  cx=0 => no crossing over boundary
                                        ;  cx<>0 => no. of bytes up to next
                                        ;           64K boundary
        je      no_crossover
        push    ax                      ;save byte count of packet
        push    bx                      ;save address offset of packet
        call    Set_PC_DMA_address      ;set up DMA address of 8237
        mov     ax,cx                   ;ax = no. of bytes that fit in present
                                        ;      64k segment
        call    dma2nic                 ;DMA 1st part of packet
        pop     ax                      ;get back original byte count
        pop     bx                      ;get address of packet
        add     bx,cx                   ;bx = address offset of next 64k
                                        ;      segment
        call    Set_PC_DMA_address      ;set up DMA address of 8237
        sub     ax,cx                   ;ax = the remaining bytes  that
                                        ;      reside in the next 64k segment
        call    dma2nic                 ;DMA 2nd part of packet
        jmp     tx_packet

no_crossover:                           ;If here there is no crossing over
                                        ;  a 64k boundary
        call    Set_PC_DMA_address      ;sets the page and address
        call    dma2nic                 ;DMA it out to the NIC card

tx_packet:
        mov     dx,COMMAND
        mov     al,26h                  ;write the transmit command
        out     dx,al
        jmp     Finished

QueueIt:
        call    Queue_packet            ;this routine queues the packet
                                        ; in memory.  Since there various
                                        ; methods to implement a queue, the
                                        ; details are not shown.

Finished:
        sti                             ;enable interrupts
        ret

DriverSend      endp
```

```
;••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
;                       Set_PC_DMA_address
;
; Converts segment and offset address of the packet to an
; absolute 20 bit address.  The lower 16 bits are loaded
; into the 8237 and the upper 4 bits a loaded into the 4 bit
; latch on the PC mother board  (location = 83h).
;
;     Entry:   ds:bx = address offset of packet to be DMAed
;     Assumes: packet is in data segment (ds)
;
;••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
Set_PC_DMA_address      proc    near
        public  Set_PC_DMA_address

        mov     ax,ds                   ;get the segment
        mov     cl,4                    ;load the shift count
        rol     ax,cl                   ;shift it
        mov     ch,al                   ;save segment low byte
        and     al,0f0h                 ;zero low nybble
        add     ax,bx                   ;add shifted-segment and offset
        jnc     SetPage
        inc     ch                      ;carry means increment (next page)

SetPage:
        out     0ch,al                  ;clear byte flip-flop of 8237
        out     02h,al                  ;8237 address (l) register
        mov     al,ah                   ;get high address
        out     02h,al                  ,8237 address (h) register
        mov     al,ch
        and     al,0fh
        out     83h,al                  ;set 8237 page register
        ret

Set_PC_DMA_address      endp

;••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
;                       dma2nic
;
; DMAs the packet from  the IBM PC to the NIC card.
;
;     Entry:   ax = byte count of packet
;     Assumes: 8237 DMA address set up
;
;••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
dma2nic proc    near
        public  dma2nic

        push    ax                      ;save the count
        mov     dx,REMOTEBYTECOUNT0
        out     dx,al                   ;NIC remote DMA count (l)
        mov     al,ah
        mov     dx,REMOTEBYTECOUNT1
        out     dx,al                   ;NIC remote DMA count (h)
        pop     ax
        dec     ax                      ;no. of bytes to xfer for 8237 =
                                        ;    byte count - 1
        out     0ch,al                  ;reset byte flip-flop of 8237
        out     03h,al                  ;8237 DMA counter (l)
        mov     al,ah
        out     03h,al                  ;8237 DMA counter (h)
        mov     al,49h
        out     0bh,al                  ;set up 8237 to do DMA read
        mov     al,1
        out     0ah,al                  ;unmask channel 1 of 8237
        mov     al,12h                  ;NIC remote DMA write command
        mov     dx,COMMAND
        out     dx,al                   ;out to the NIC command register
        mov     dx,INTERRUPTSTATUS

wait:
        in      al,dx
        test    al,40h                  ;wait for RDC bit in ISR to go high
        jz      wait
        mov     al,5
        out     0ah,al                  ;mask out channel 1 of 8237

        ret

dma2nic endp

;•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
;  PageWrapCheck - checks to see if the DMA will crossover a
;                  64k byte boundary.
;
;       Entry:  ds:bx = address offset of packet
;               ax    = byte count of packet
;
;       Returns: cx =  0  => packet fits in present 64k segment
;                cx <> 0  => number of bytes that will fit in present
;                            64k segment
;
;•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
PageWrapCheck proc near
        public   PageWrapCheck

        push    ax                      ;save byte count
        push    bx                      ;save address offset
        mov     dx,ds                   ;dx = data segment
        mov     cl,4                    ;cl = shift left counter
        shl     dx,cl
        add     dx,bx                   ;dx = lower 16 bits of absolute addr.
        add     ax,dx                   ;check if packet crosses over
        jnc     no_wrap                 ;jump if no 64k crossover
        mov     cx,0
        sub     cx,dx                   ;cx=no. of bytes that fits in segment
        pop     bx
        pop     ax
        ret

no_wrap:
        mov     cx,0                    ;return code => no crossover
        pop     bx
        pop     ax
        ret

PageWrapCheck endp

Code    ends
        end
```

```
;•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
;                          DriverISR
;
; This interrupt service routine responds transmit, transmit error, and
; receive interrupts (the PTX, TXE, and PRX bits in the INTERRUPT STATUS
; register) produced from the NIC.  Upon transmit interrupts, the upper
; layer software is informed of successful or erroneous transmissions;
; upon receive interrupts, packets are removed from the Receive Buffer
; Ring (in local memory) and transfer to the PC.
;
;•••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••

;••••••••••••••••••  Equates for NIC registers  •••••••••••••••••••

COMMAND                 equ     2e0h
PAGESTART               equ     COMMAND+1
PAGESTOP                equ     COMMAND+2
BOUNDARY                equ     COMMAND+3
TRANSMITSTATUS          equ     COMMAND+4
TRANSMITPAGE            equ     COMMAND+4
TRANSMITBYTECOUNT0      equ     COMMAND+5
NCR                     equ     COMMAND+5
TRANSMITBYTECOUNT1      equ     COMMAND+6
INTERRUPTSTATUS         equ     COMMAND+7
CURRENT                 equ     COMMAND+7           ;in page 1
REMOTESTARTADDRESS0     equ     COMMAND+8
CRDA0                   equ     COMMAND+8
REMOTESTARTADDRESS1     equ     COMMAND+9
CRDA1                   equ     COMMAND+9
REMOTEBYTECOUNT0        equ     COMMAND+0ah
REMOTEBYTECOUNT1        equ     COMMAND+0bh
RECEIVESTATUS           equ     COMMAND+0ch
RECEIVECONFIGURATION    equ     COMMAND+0ch
TRANSMITCONFIGURATION   equ     COMMAND+0dh
FAE_TALLY               equ     COMMAND+0dh
DATACONFIGURATION       equ     COMMAND+0eh
CRC_TALLY               equ     COMMAND+0eh
INTERRUPTMASK           equ     COMMAND+0fh
MISS_PKT_TALLY          equ     COMMAND+0fh

PSTART                  equ     26h
PSTOP                   equ     40h

CGroup  group   Code

Code    segment para public 'Code'

        assume  cs:CGroup,ds:CGroup,es:nothing,ss:nothing

;  External routines

      extrn  DriverSend:          near
      extrn  Set_PC_DMA_address: near

;  Public domain variables

      public  recv_pc_buff
      public  headptr
      public  tailptr
```

```
; Declaration of variables

recv_pc_buff    db      2048 dup (?)    ;buffer for holding recved packet
headptr         dw      ?               ;head ptr for tx Q
tailptr         dw      ?               ;tail ptr for tx Q
byte_count      dw      ?
imr             db      1bh             ;image of Interrupt Mask register

;*****************************************************************
;  Begin of Interrupt Service Routine
;*****************************************************************
netisr proc near
       public netisr

       cli
       push   ax                        ;save regs
       push   bx
       push   cx
       push   dx
       push   di
       push   si
       push   ds
       push   es
       push   bp

       mov    al,0bch
       out    21h,al                    ;turn off IRQ3
       sti

       mov    ax,CGroup
       mov    ds,ax                     ;ds=cs

;-----------------------------------------------------------------
;  Read INTERRUPT STATUS REGISTER for receive packets, transmitted
;  packets and  errored transmitted packets.
;-----------------------------------------------------------------

poll:
       mov    dx,INTERRUPTSTATUS
       in     al,dx
       test   al,1                      ;packet received?
       jnz    pkt_recv_rt
       test   al,0ah                    ;packet transmitted?
       jz     exit_isr                  ;no, let's exit
       jmp    pkt_tx_rt

exit_isr:
       mov    dx,INTERRUPTMASK          ;disabling NIC's intr
       mov    al,0
       out    dx,al
       cli
       mov    al,0b4h                   ;turn IRQ3 back on
       out    21h,al
       mov    al,63h                    ;send 'EOI' for IRQ3
       out    20h,al
       sti

       mov    dx,INTERRUPTMASK          ;NOTE: intr from the NIC
       mov    al,imr                    ; are enabled at this point so
       out    dx,al                     ; that the 8259 interrupt
                                        ; controller does not miss any
                                        ; IRQ edges from the NIC
                                        ; (IRQ is edge sensitive)
```

```
        pop     bp
        pop     es
        pop     ds
        pop     si
        pop     di
        pop     dx
        pop     cx
        pop     bx
        pop     ax
        iret
;------------------------------------------------------------------
;  Packet Receive Routine (pkt_recv_rt) -- clears out all good
;  packets in local receive buffer ring. Bad packets are ignored.
;------------------------------------------------------------------

pkt_recv_rt.
        mov     dx,INTERRUPTSTATUS
        mov     al,1
        out     dx,al                   ;reset PRX bit in ISR
        call    dma2pc                  ;DMA pkt from NIC to PC

;    ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;
;    Inform upper layer software of a received packet to be processed
;
;    ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

;  checking to see if receive buffer ring is empty

        mov     dx,BOUNDARY
        in      al,dx
        mov     ah,al                   ;save BOUNDARY in ah
        mov     dx,COMMAND
        mov     al,62h
        out     dx,al                   ;switched to pg 1 of NIC
        mov     dx,CURRENT
        in      al,dx
        mov     bh,al                   ;bh = CURRENT PAGE register
        mov     dx,COMMAND
        mov     al,22h
        out     dx,al                   ;switched back to pg 0
        cmp     ah,bh                   ,recv buff ring empty?
        jne     pkt_recv_rt
        jmp     poll

;------------------------------------------------------------------
;  packet_transmit_routine (pkt_tx_rt) -- determine status of
;  transmitted packet, then checks the transmit-pending
;  queue for the next available packet to transmit.
;------------------------------------------------------------------

pkt_tx_rt:
        mov     dx,INTERRUPTSTATUS
        mov     al,0ah
        out     dx,al                 ;reset PTX and TXE bits in ISR

        mov     dx,TRANSMITSTATUS     ;check for erroneous TX
        in      al,dx
        test    al,38h                ;is FU, CRS, or ABT bits set in TSR
        jnz     bad_tx
```

```
;  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;
;  Inform upper layer software of successful transmission
;
;  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
        jmp     chk_tx_queue

bad_tx:                                         ;in here if bad TX

;  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
;
;  Inform upper layer software of erroneous transmission
;
;  ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

chk_tx_queue:
        mov     dx,headptr
        mov     ax,tailptr
        cmp     ax,dx                   ;No packets in TX queue?
        je      poll

        mov     si,tailptr              ;si points to the next packet to
        call    DriverSend              ;  be transmitted.
                                        ;  (entry for DriverSend)

;  Update tailptr to next packet in queue

        mov     si,tailptr              ;link field is at the beginning
                                        ;    of each packet
        mov     dx,[si]                 ;go to link field of packet
        mov     tailptr,dx              ;move tailptr to next packet
        jmp     poll

netisr endp

;••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••
;  dma2pc - this routine DMAs the next packet in the Receive
;           Buffer Ring and places into the IBM PC memory.
;
;           Entry:  none
;
;••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••••

dma2pc proc near
        public dma2pc
```

SECTION 5

APPLICATION NOTES

DP8390 Introductory Guide
HARDWARE DESIGN HINTS
LOOPBACK GUIDE
STARLAN

DP8390 Network Interface Controller: An Introductory Guide

National Semiconductor Corp.
Application Note 475

OVERVIEW

A general description of the DP8390 Network Interface Controller (NIC) is given in this application note. The emphasis is placed on how it operates, and how it can be used. This description should be read in conjunction with the DP8390 data sheet.

1.0 INTRODUCTION

The DP8390 Network Interface Controller provides all the Media Access Control layer functions required for transmission and reception of packets in accordance with the IEEE 802.3 CSMA/CD standard. The controller was designed to act as an advanced peripheral and serve as a complete interface between the system and the network. The onboard FIFO and DMA channels work together to form a straight-forward packet management scheme, providing (local) DMA transfers at up to 10 megabytes per second while tolerating typical bus latencies.

A second set of DMA channels (remote DMA) is provided on chip, and is integrated into the packet management scheme to aid in the system interface. The DP8390 was designed with the popular 8, 16 and 32 bit microprocessors in mind, and gives system designers several architectural options. The NIC is fabricated using National Semiconductor's double metal 2 micron microCMOS process, yielding high speed with very low power dissipation.

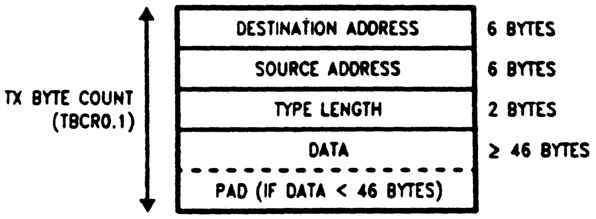
2.0 METHOD OF OPERATION

The NIC is used as a standard peripheral device and is controlled through an array of on-chip registers. These registers are used during initialization, packet transmission and reception, and remote DMA operations. At initialization, the physical address and multicast filters are set, the receiver, transmitter and data paths are configured, the DMA channels are prepared, and the appropriate interrupts are masked. The Command Register (CR) is used to initiate transmission and remote DMA operations.

Upon packet reception, end of packet transmission, remote DMA completion or error conditions, an interrupt is generated to indicate that an action should be taken. The processor's interrupt driven routine then reads the Interrupt Status Register (ISR) to determine the type of interrupt that occurred, and performs the appropriate actions.

3.0 PACKET TRANSMISSION

The NIC transmits packets in accordance with the CSMA/CD protocol, scheduling retransmission of packets up to 15 times on collisions according to the truncated binary exponential backoff algorithm. No additional processor intervention is required once the transmit command is given.

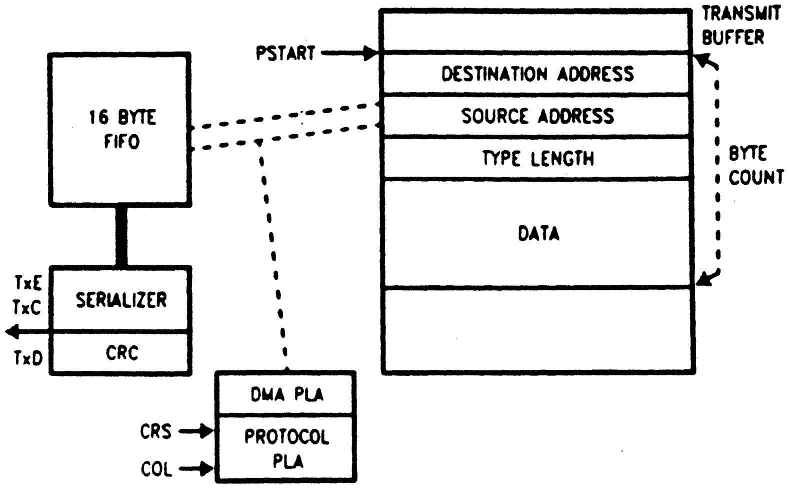


TL/F/9141-1

FIGURE 1. Transmit Packet Format

3.1 Transmission Setup

After a packet that conforms to the IEEE 820.3 specification is set up in memory, with 6 bytes of the destination address, followed by 6 bytes of the source address, followed by the data byte count and the data, it is ready for transmission (see *Figure 1*). To transmit a packet, the NIC is given the starting address of the packet (TPSR), the length of the packet (TPCR0, TBCR1), and then the PTX (transmit packet) bit of the Command Register is set to initiate the transmission (see *Figure 2*).



TL/F/9141-2

FIGURE 2. Packet Transmission

DP8390 Network Interface Controller: An Introductory Guide

AN-475

3.2 Transmission Process

Once the transmit command is given, if no reception is in progress, the transmit prefetch begins. The high speed local DMA channel bursts data into the NIC's FIFO. After the first DMA transfer of the prefetch burst, if no carrier is present on the network, and the NIC is not deferring, the TXE (transmit enable) signal is asserted and the transmission begins. After the 62 bits of preamble (alternating ONEs and ZEROs) and the start of frame delimiter (two ONEs) are sent out, the data in the FIFO is serialized, and sent out as NRZ data (pin TxD) with a clock (TxC), while the CRC is calculated. When the FIFO reaches a threshold (X bytes empty) a new DMA burst is initiated. This process continues until the byte count reaches zero. After the last byte is serialized, the four bytes of the calculated CRC are serialized and appended to complete the packet.

Should a collision occur, the current transmission stops, a jam sequence (32 Ones) transmitted (to ensure that every node senses a collision), and a retransmission of the packet is scheduled according to the truncated Binary Exponential Backoff Routine.

3.3 Transmission Status

After the transmission is complete, an interrupt is generated and either the PTX bit (complete packet transmitted) or the TXE bit (packet transmission aborted) of the ISR (Interrupt Status Register) is set. The interrupt driven routine then reads the TSR to find out details of the transmission. If the PTX bit is set, the TSR can reveal if a carrier was present when the transmission was initiated (DFR), if the carrier was lost during the transmission (CRS—this would point to a short somewhere on the network), if the collision detect circuitry is working properly (CDH), and if collision occurred (COL). Whenever a collision is encountered during transmission, the collision count register (NCR) is incremented. Should a collision occur outside the 512 bit window (slot time), the OWC (Out of Window Collision) bit of the TSR is set.

The TXE bit of the ISR is set if 16 collisions or a FIFO underrun occurs. If the transmission is aborted due to 16 collisions, the ABT bit of the TSR is set. (If this occurs it is likely that there is an open somewhere on the network.) If the local DMA channel can not fill the FIFO faster than data is sent to the network, the FU bit (FIFO Underrun) of the TSR is set and the transmission is also aborted. This is a result of a system bandwidth problem and points to a system design flaw. System bandwidth considerations are discussed further in Section 5.1.3.

4.0 PACKET RECEPTION

The bus topology used in CSMA/CD networks allows every node to receive every packet transmitted on the network. The receive filters determine which packets will be buffered to memory. Since every packet is not of interest, only packets having a destination address that passes the node's receive filters will be transferred into memory. The NIC offers many options for the receive filters and implements a complete packet management scheme for storage of incoming packets.

4.1 Reception Process

When a carrier is first sensed on the network (i.e. CRS signal is active), the controller sees the alternating ONE - ZERO preamble and begins checking for two consecutive ONEs, denoting the start of frame delimiter (SFD). Once the SFD is detected, the serial stream of data is deserialized and pushed into the FIFO, a byte at a time. As the data is being transferred into the FIFO, the first six bytes are checked against the receive address filters. If an address match occurs, the packet is DMAed from the FIFO into the receive buffer ring. If the address does not match, the packet is not buffered and the FIFO is reset.

Each time the FIFO threshold is reached, a DMA burst begins and continues for the proper number of transfers. DMA bursts continue until the end of the packet (Section 5.1.2). At the end of a reception, the NIC prepares for an immediate reception while writing the status of the previous reception to memory. An interrupt is issued to indicate that a packet was received, and is ready to be processed.

The CRC generator is free running and is reset whenever the SFD is detected. At every byte boundary the calculated value of the CRC is compared with the last four received bytes. When the CRS signal goes LOW, denoting the end of a packet, if the calculated CRC matches the received CRC on the last byte boundary, the packet is a good packet and is accepted. However, if the calculated and received CRCs do not match on the last byte boundary before CRS goes LOW, a CRC error is flagged (CRC bit of RSR set) and the packet is rejected, i.e. the receive buffer ring pointer (CURR) is not updated (Section 4.5). If the CRS signal does not go LOW on a byte boundary and a CRC error occurs, the incoming packet is misaligned, and a frame alignment error is flagged (FAE bit of RSR set). Frame alignment errors only occur with CRC errors.

4.2 Address Matches

The first bit received after the SFD indicates whether the incoming packet has a physical or multicast address. A ZERO indicates a physical address, that is, a unique map-

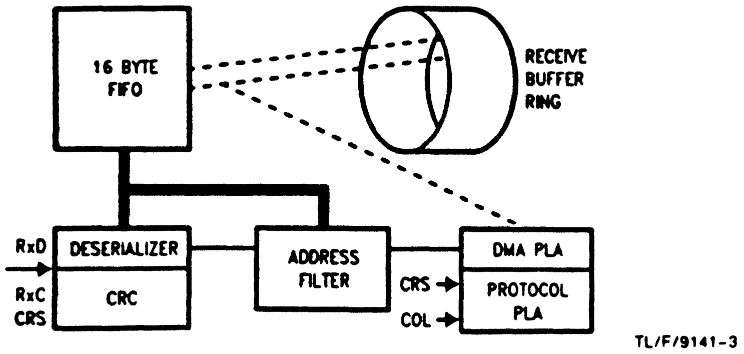


FIGURE 3. Packet Reception

ping between the received address and the node's 48 bit physical address as programmed at intialization (PAR0–PAR5). A ONE indicates a multicast address, meaning a packet intended for more than one node.

Multicast addressing is useful where one node needs to send a packet to multiple nodes, as in a query command. Multicast addressing provides a very fast way to perform address filtering in realtime, by using an on-chip hashing table. A hashing algorithm based on the CRC is used to map the multicast address into the 64 bit Multicast Address Filter (MAF0–7).

After the CRC has been calculated on the destination address, the upper six bits of the CRC are used as an index into the Multicast Address Filter (MAF). If the selected filter bit is ONE, the packet is accepted, if the MAF bit is ZERO the packet is not accepted.

A special multicast address is the broadcast address, which denotes a packet intended to be received by all nodes. The broadcast packet has an address of all ONEs (this address also maps into a bit in the MAF).

The DP8390 also provides the ability to accept all packets on the network with a physical address. Promiscuous mode causes any packet with a physical address to be buffered into memory. To receive all multicast packets it is nesessary to set all of the MAF bits to ONE.

4.3 Network Statistics

Three eight bit counters are provided for monitoring receive packet errors. After an address match occurs if a Frame Alignment or CRC error occurs, or if a packet is lost due to insufficient buffer resources (see below), the appropriate counter is incremented. These counters are cleared when read. The counters trigger an interrupt when they reach a value of 128 (if not masked) to force the processor to read (and thus clear) their contents. The counters have a maximum value of 192, providing a large latency between when the interrupt is asserted and when the counter overflows. When a CNT interrupt occurs, all three tally counters should be read and added into larger counters maintained by the processor.

4.4 Setting the Receive Configuration Register

The Receive Configuration Register (RCR) is used in conjunction with the physical and multicast addresses to determine which packets should be accepted and placed in the receive buffer ring. The RCR is initialized to accept physical, multicast and/or broadcast packets, or alternatively to place the receiver in promiscuous mode to accept all packets with a physical address. If the MON bit of the RCR is set, placing the receiver in monitor mode, the receiver still checks the addresses of incoming packets according to the set up address filter, and network statistics are still gathered, but packets are not buffered into memory.

The minimum packet size in standard 802.3 networks is 64 bytes long. Packets less than 64 bytes are considered runt packets and are normally rejected. However, in some applications it may be desirable to accept such packets. By setting the AR bit of the RCR, runt packets are accepted.

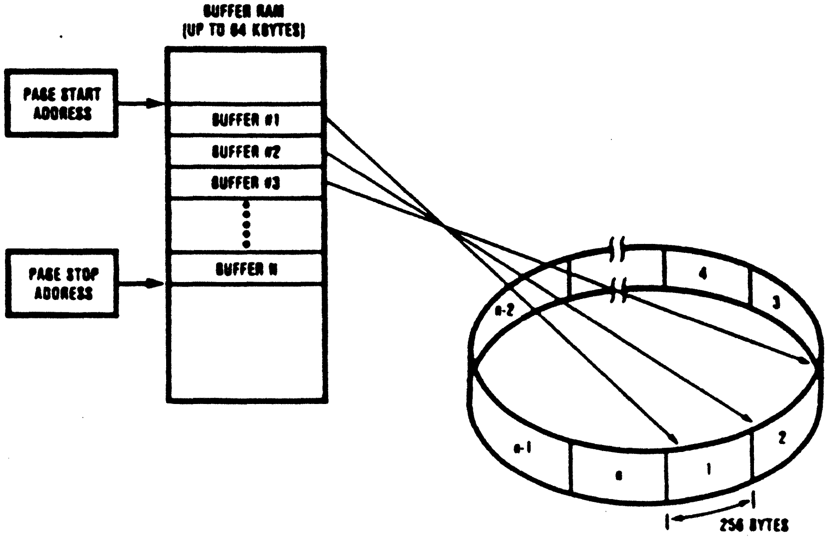
For diagnostic purposes it may be desirable to examine errored packets, and not overwrite them with good packets as is done in normal operation. By setting the SEP bit of the RCR, errored packets are saved and their status is written to memory.

4.5 Receive Buffer Ring

As packets are received they are placed into the receive buffer ring, and as they are processed they are removed from this ring. At initialization, an area of memory is allocated to act as the receive buffer ring, and the NIC's buffer management scheme then makes efficient use of this memory. The efficiency is helped significantly because the ring pointers are contained on chip, and the DMA channels can work at up to a 10 Mbyte/sec transfer rate. A second DMA channel, the remote DMA channel, is available for transferring packets out of the receive buffer ring.

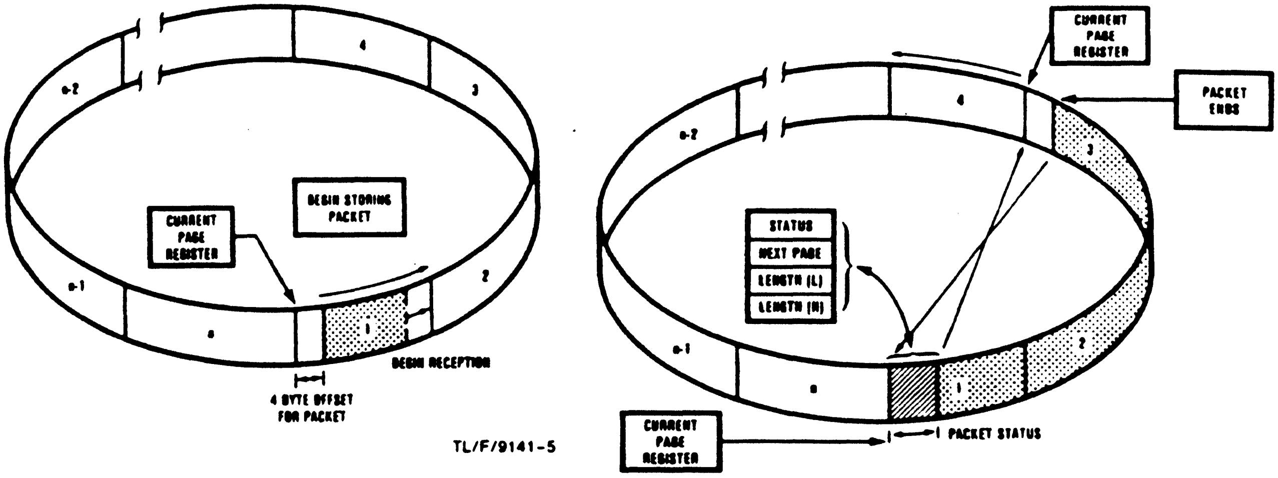
The employed buffer management scheme effectively works as a large packet FIFO. This buffer management scheme is very appropriate for most networking applications because packets are generally processed in the order they are received.

Four pointers are used to control the ring; the (1) page start (PSTART) and (2) page stop (PSTOP) pointers to determine the size of the buffer ring, the (3) current page (CURR) pointer, to determine where the next packet will be loaded,



TL/F/9141-4

FIGURE 4. The Receive Buffer



TL/F/9141-5

TL/F/9141-6

FIGURE 5. Receive Packet Buffering

and the (4) boundary (BNRY) pointer, to show where the next packet to be unloaded (or processed) lies. As packets are received, the boundary pointer follows the current page pointer around the ring. The page start and stop pointers remain unchanged during operation.

The receive buffer ring is divided into 256 byte buffers, and these buffers are linked together as required by the received packets (see *Figure 4*). Up to 256 of these buffers can be linked together in the receive buffer ring, yielding a maximum buffer size of 64K bytes. Since all NIC registers are 8 bits wide, the ring pointers refer to 256 byte boundaries within a 64K byte space.

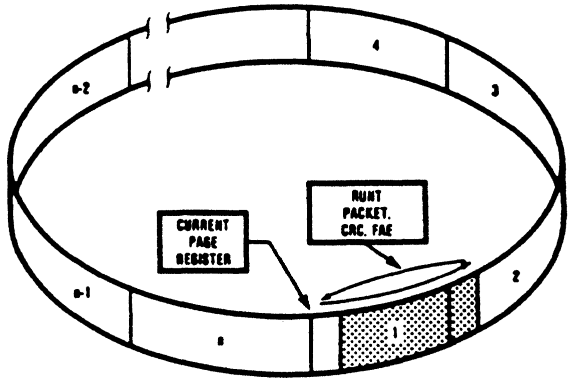
At initialization, PSTART register is loaded with the beginning page address of the ring, and PSTOP is loaded with the ending page address of the ring.

On a valid reception, the packet is placed in the ring at the page pointed to by CURR plus a 4 byte offset (see *Figure 5*). The packet is transferred to the ring, a DMA burst at a time. When necessary, buffers are automatically linked together, until the complete packet is received. The last and first buffers of the ring buffer are linked just as the first and seconds buffers. At the end of a reception, the status from the Receive Status Register (RSR), a pointer to the next packet, and the byte count of the current packet are written into the 4 byte offset.

If a receive error occurs (FAE, CRC) CURR is not updated at the end of a reception, so the next packet received overwrites the bad packet (see *Figure 6*). This feature can be disabled (by setting the save errored packet (SEP) bit in the RCR) to allow examination of errored packets.

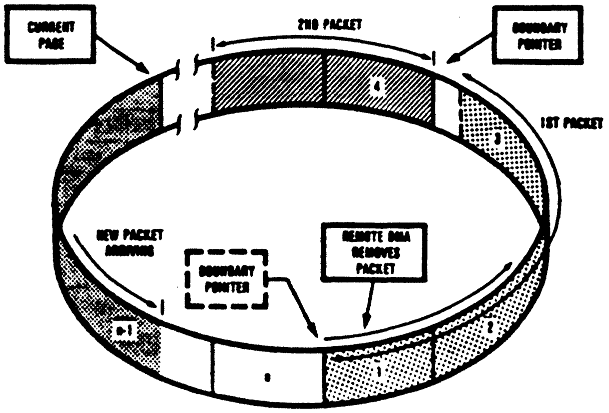
At receiving nodes, collision fragments may be seen as runt packets. A runt packet is a packet less than 64 bytes (512 bits) long, and since a collision must occur in the first 512 bit times, the packet will be truncated to less than 64 bytes. After runt packets are received, the CURR is not updated, so the next packet received will overwrite the runt packet. This standard feature can also be suppressed by setting the AR bit in the TCR. This is useful when it is desirable to examine collision fragments, and in non-standard applications where smaller packets are desirable.

Once packets are in the receive ring they must be processed. However, the amount of processing that occurs while the packet is in the buffer ring varies according to the implementation. As packets are removed from the buffer ring, the boundary pointer (BNRY) must be updated. The BNRY always follows CURR around the ring (see *Figure 7*).



TL/F/9141-7

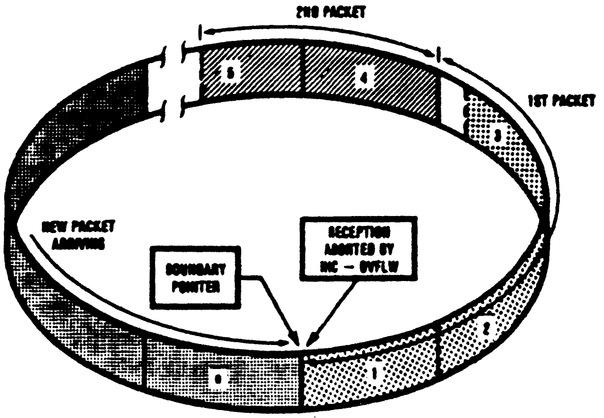
FIGURE 6. Packet Rejection



TL/F/9141-8

FIGURE 7. Removing Packets From Receive Buffer Ring

If the current local DMA address ever reaches BNRY, the ring is full. In this case, the current and any additional receptions are aborted and tallied until the BNRY pointer is updated. Packets already present in the ring will not be overwritten (see *Figure 8*). All missed packets will increment the missed packet tally counter. When enough memory is allocated for the receive buffer ring, the overwrite warning (setting of the OVW bit of the ISR) should seldom occur.



TL/F/9141-9

FIGURE 8. Receive Buffer Ring Overwrite Protection

A second set of DMA channels has been included on the DP8390 to aid in the transfer of packets out of the buffer ring. These Remote DMA channels can work in close co-operation with the receive buffer ring to provide a very effective system interface (§7).

If the BNRY is placed outside of the buffer ring, no overwrite protection will be present, and incoming packets may overwrite packets that have not been processed. This may be useful when evaluating the DP8390, but in normal operation it is not recommended.

When the CURR and BNRY pointers are equal, the buffer ring can either be completely empty or completely full. To ensure that the NIC does not misinterpret this condition, it is necessary to guarantee that the value of the BNRY pointer does not equal the value of the CURR pointer. It is recommended that the BNRY pointer be kept one less than CURR pointer when the ring is empty, and only be equal to CURR when the ring is full, as shown below.

1. Use a variable (NXTPKT) to indicate from where the next packet will be removed (possibly using Remote DMA)
2. At initialization set:
   BNRY = PSTART
   CURR = PSTART + 1
   NXTPKT = PSTART + 1
3. After each packet is removed from the ring, use the next packet pointer in the header information (the second byte of the header), HNXTPKT, and set:
   NXTPKT = HNXTPKT
   BNRY = HNXTPKT − 1
   If BNRY < PSTART then BNRY = PSTOP − 1

The above procedure is not necessary if the Send Packet Command is used to remove packets from the ring as explained in section 7.

5.0 SYSTEM/NETWORK INTERFACE

The DP8390 offers considerable flexibility when designing a system/network interface. This flexibility allows the designer to choose the appropriate price/performance combination while easing the actual design process.

5.1 Interfacing Considerations

Several features have been included on the NIC to allow it to easily be integrated into many systems. The size of the data paths, the byte ordering, and the bus latencies are all programmable. In addition, the clock the DMA channels use is not coupled to the network clock, so the NIC's DMA can easily be integrated into memory systems.

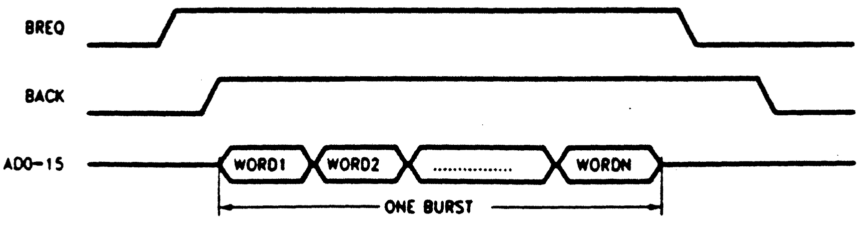
5.1.1 Data Path

The NIC can interface with 8, 16, and 32 bit microprocessors. The data paths are configurable for both byte- wide and word-wide transfers (bit WTS in DCR). When in word-wide mode, the byte ordering is programmable to accommodate both popular byte ordering schemes. All NIC registers are 8 bits wide to allow 8, 16 and 32 bit processors to access them with no additional hardware. If the NIC's 16 address lines (64K bytes) do not provide an adequate address space, the two DMA channels can be concatenated to form a 32 bit DMA address (bit LAS in DCR).

5.1.2 Local DMA

The DMA transfers between the FIFO and memory during transmission and reception occur in bursts. The bursts begin when the FIFO threshold is reached. Since only a single FIFO is required (because a node cannot receive and transmit simultaneously), the threshold takes on different meanings during transmission and reception. During reception the FIFO threshold refers to the number of bytes in the FIFO. During transmission the FIFO threshold refers to the number of empty bytes in the FIFO (16 - # bytes in FIFO). The FIFO threshold is set to 2, 4, 8 or 12 bytes (1, 2, 4 or 6 words) in the DCR (bits FT0, FT1).

The number of transfers that occur in a burst depends on whether the Exact Transfer or Empty/Fill mode is used (bit BMS in DCR). When in Exact Transfer mode, a number of bytes/words equal to the FIFO threshold will be transferred in each burst. The Empty/Fill mode continues the transfers until the FIFO is empty, during receptions, and full, during transmissions (see *Figure 8*).



TL/F/9141-10

where N = 1, 2, 4 or 6 Words or N = 2, 4, 8, or 12 Bytes when in byte mode

FIGURE 8. Local DMA Burst

Before a burst can begin, the NIC must first arbitrate to become master of the bus. It requests the bus by activating the BREQ signal and waiting for acknowledgment with the BACK signal. Once the NIC becomes the master of the bus, the byte/word transfers may begin. The frequency of the DMA clock is not related to the network clock, and can be input (pin 25) as any frequency up to 20 MHz. For 10 Mbit/sec networks the DMA clock can be as slow as 6 MHz. This allows tailoring of the DMA channel, to the system. The local DMA channel can burst data into and out of the FIFO at up to 10 Mbyte/sec (8X the speed of standard Ethernet). This means that during transmission or reception the network interface could require as little as one eighth of the bus bandwidth.

5.1.3 Bus Analysis

Two parameters useful in analysis of bus systems are the Bus Latency and the Bus Utilization. The Bus Latency is the maximum time between the NIC assertion of BREQ and the system granting of BACK. This is of importance because of the finite size of the NIC's internal FIFO. If the bus latency becomes too great, the FIFO overflows during reception (FIFO overrun error), and becomes empty during transmission (FIFO underrun error). Both conditions result in an error that aborts the reception or transmission. In a well designed system these errors should never occur. The Bus Utilization is the fraction of time the NIC is the master of the bus. It is desirable to minimize the time the NIC occupies the bus, in order to maximize its use by the rest of the system. When designing a system it is necessary to guarantee the NIC a certain Bus Latency, and it is desirable to minimize the Bus Utilization required by the NIC.

Associated with each DMA burst is a DMA set up and recovery time. When a packet is being transferred either to or from memory it will be transferred in a series of bursts. If more byte/word transfers are accomplished in each burst, fewer bursts are required to transfer the complete packet, and less time is spent on DMA set up and recovery. Thus, when longer bursts are used, less bus bandwidth is required to complete the same packet transfer.

The Empty(/Fill) mode guarantees longer bursts because as the byte/word transfers are taking place, serialized data is still filling(/emptying) the FIFO, and these additional bytes/words must also be transferred out of(/into) the FIFO. The least NIC bus utilization occurs when the bursts are as long as possible. This occurs when the threshold is as high as possible, and Empty/Fill mode is used. The determination of the threshold is related to the maximum bus latency the system can guarantee the NIC.

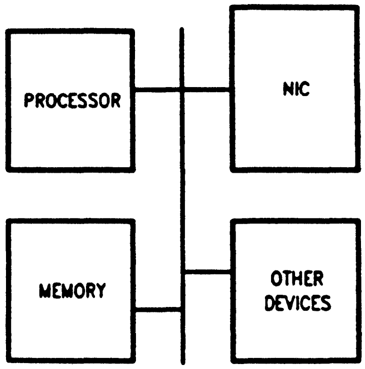
If the NIC is required to guarantee other devices a certain bus latency, it can only remain master of the bus for a certain amount of time. In this case, the Exact Transfer burst mode is desirable because the NIC only remains master of the bus for a certain amount of time.

6.0 INTERFACE OPTIONS

The network interface can be incorporated into systems in several ways. The network interface can be controlled by either a system processor or a dedicated processor, and can utilize either system memory or buffer memory. This section covers the basic interface architectures.

6.1 Single Bus System

The least complex implementation places the NIC on the same bus as the processor (see *Figure 10*). The DP8390 acts as both a master and a slave on this bus; a master during DMA bursts, and a slave during NIC register accesses. This architecture is commonly seen on motherboards in personal computers and low cost workstations, but until recently without an integrated network interface. A major issue in such designs is the bus bandwidth for use by the processor. The DP8390 is particularly suitable for such applications because of its bus utilization characteristics. During transmissions and receptions, the only time the NIC becomes a bus master, the DP8390 can require as little as one- eighth the bus bandwidth. In addition, the previously mentioned bus tailoring features, ease its integration into such systems.



TL/F/9141-11

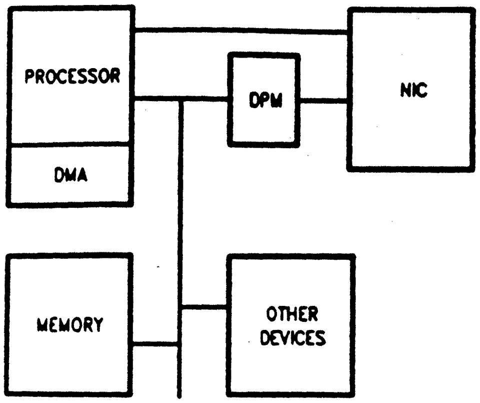
FIGURE 10. Single Bus Configuration

The design must only be able to guarantee the NIC a maximum bus latency (< 9 μs for 10 Mbit/s networks), because of the finite size of the on-chip FIFO. In bus systems where the NIC is the highest priority device, this should present no problem. However, if the bus contains other devices such as Disk, DMA and Graphic controllers that require the bus for more than 10 μs during high priority or real time activities, meeting this maximum bus latency criteria could present a problem.

Likewise, many existing single bus systems make no provision for external devices to become bus masters, and if they do, it is only under several restrictions. In such cases, an interface without the mentioned bus latency restrictions is highly desirable.

6.2 Dual Port Memory

One popular method of increasing the apparent bus latency of an interface, has the added effect of shielding the system bus from the high priority network bandwidth. In this application, the Dual Port Memory (DPM) allows the system bus to access the memory through one port, while the network interface accesses it through the other port. In this way, all of the high priority network bandwidth is localized on a dedicated bus, with little effect on the system bus (see *Figure 11*).



TL/F/9141-12

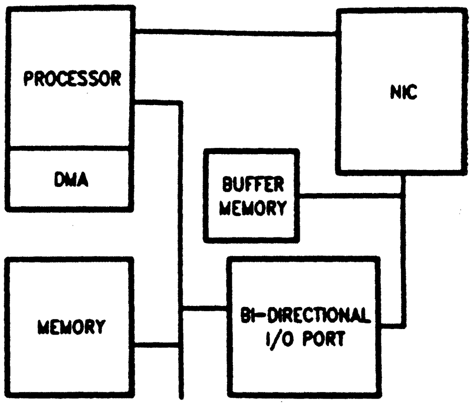
FIGURE 11. DPM Configuration

Dual Port Memories are typically smaller than the main memory and little, if any, processing can occur while the packets are in the DPM. Therefore, the processor (or if available, DMA controller) must transfer data between the DPM and the main memory before beginning packet processing. In this example, the DPM acts as a large packet FIFO.

Such configurations provide workable solutions, however, Dual Port Memories are inherently expensive. Aside from the extra complexity of the software, DPM contention logic is expensive, and dedicated DPM chips provide only 1k of memory and cost as much as advanced VLSI devices. In addition, some systems do not contain additional memory for such memory mapped interfaces.

6.3 Dual Port Memory Equivalent

The functional equivalent of a Dual Port Memory implementation can be realized for low cost with the DP8390. This configuration makes use of the NIC's Remote DMA capabilities and requires only a buffer memory, and a bidirectional I/O port (see *Figure 12*). The complete network interface, with 8k x 8 of buffer memory, easily fits onto a half size IBM-PC card (as in the Network Interface Adapter, NIA, for the IBM-PC.)



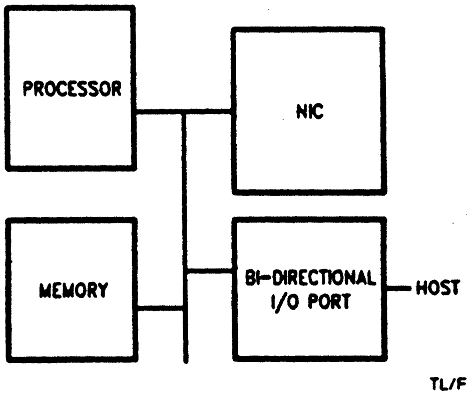
TL/F/9141-13

FIGURE 12. DPM Equivalent Configuration

The high priority network bandwidth is decoupled from the system bus, and the system interracts with the buffer memory using a lower priority bi-directional I/O port. For example, when a packet is received the local DMA channel transfers it into the buffer memory, part of which has been configured as the receive buffer ring. The remote DMA channel then transfers the packet on a byte by byte (or word by word) basis to the I/O port. At this point, as in the previous example, the processor (or if available, DMA channel), through a completely asynchronous protocol, transfers the packet into the main memory.

6.4 Dual Processor Configuration

For higher performance applications, it is desirable to offload the lower-level packet processing functions from the main system (see *Figure 11*). A processor placed on a local bus with the NIC, memory and a bi-directional I/O port could accomplish these lower-level tasks, and communicate with the system processor through a higher level protocol. This processor could be responsible for sending acknowledgement packets, establishing and breaking logical links, assembling and disassembling files, executing remote procedure calls, etc.



TL/F/9141-14

FIGURE 13. Dual Processor Configuration

7.0 REMOTE DMA

A second set of DMA channels is built into the DP8390 to aid in the system integration (as discussed above). Using a simple asynchronous protocol, the Remote DMA channels are used to transfer data between dedicated network memory, and common system memory. In normal operation, the remote DMA channels transfer data between the network memory and an I/O port, and the system transfers between the I/O port and the system memory. The system transfers are typically accomplished using either the processor, or a DMA controller.

The Remote DMA channels work in both directions; pending transmission packets are transferred into the network memory, and received packets are transferred out of the network memory. Transfers into the network memory are known as remote write operations, and transfers out of the network memory are known as remote read operations. A special remote read operation, send packet, automatically removes the next packet from the receive buffer ring.

7.1 Performing Remote DMA Operations

Before beginning a remote DMA operation, the controller must be informed of the network memory it will be using. Both the starting address (RSAR0,1) and length (RBCR0,1) are set before initiating the remote DMA operation. The remote DMA operation begins by setting the appropriate bits in the Command Register (RD0–RD3). When the remote DMA operation is complete (all of the bytes transferred), the RDC bit (Remote DMA Complete) in the ISR (Interrupt Status Register) is set and the processor receives an interrupt, whereupon it takes the appropriate action. When the Send packet command is used, the controller automatically loads the starting address, and byte count from the receive buffer ring for the remote read operation, and upon completion updates the boundary pointer (BNRY) for the receive buffer ring. Only one remote DMA operation can be active at a time.

7.2 Hardware Considerations

The Remote DMA capabilities of the NIC were designed to require minimal external components and provide a simple implementation. An eight bit bi-directional port can be implemented using just two 374 latches (see the DP8390 Hardwre Design Guide). All of the control circuitry is provided on the DP8390. In addition, bus arbitration with the local DMA is accomplished within the NIC in such a way as to not lock out other devices on the bus (see the DP8390 Datasheet).

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

National Semiconductor
Corporation
2900 Semiconductor Drive
P.O. Box 58090
Santa Clara, CA 95052-8090
Tel (408) 721-5000
TWX (910) 339-9240

National Semiconductor
GmbH
Westendstrasse 193-195
D-8000 Munchen 21
West Germany
Tel: (089) 5 70 95 01
Telex: 522772

NS Japan Ltd.
4-403 Ikebukuro,
Toshima-ku,
Tokyo 171, Japan
Tel (03) 988-2131
FAX. 011-81-3-988-1700

National Semiconductor
Hong Kong Ltd.
Southeast Asia Marketing
Austin Tower, 4th Floor
22-26A Austin Avenue
Tsimshatsui, Kowloon, H.K.
Tel. 3-7231290, 3-7243645
Cable: NSSEAMKTG
Telex: 52996 NSSEA HX

National Semiconductores
Do Brasil Ltda.
Av. Brig. Faria Lima, 830
8 Andar
01452 Sao Paulo, SP. Brasil
Tel: (55/11) 212-5066
Telex: 391-1131931 NSBR BR

National Semiconductor
(Australia) PTY, Ltd.
21/3 High Street
Bayswater, Victoria 3153
Australia
Tel: (03) 729-6333
Telex: AA32096

National
Semiconductor
Corporation

Hardware Design with the DP8390

February 1987

This document is intended for individuals designing with the DP8390 Network Interface Controller (NIC). A structured approach is outlined to organize the design process. The designer should be familiar with the Introductory Guide and Datasheet for the DP8390 before embarking on the design path.

The design process is divided into the network interface, the processor interface, the local DMA interface, and the remote DMA interface. The requirements for each interface are reviewed below.

1. The Network Interface

From the designers point of view, this is perhaps the easiest interface. The network interface up to the NIC is completely duplex. For transmission, an enable , clock and data line are used (TXE, TXC and TXD) and are generated by the NIC. For reception an enable, clock and data line are used (CRS, RXC and RXD) and generated by the manchester decoder. Should collisions occur on the network, the COL signal is asserted, causing the NIC to begin the retransmission process. A diagnostic pin is provided to facilitate loopback testing through the manchester encoder/decoder.

For Ethernet/Cheapernet applications, the NIC can be connected directly to the DP8391 Serial Network Interface (SNI), as shown below. The pinout of the NIC and the SNI has been arranged to aid in the component layout.

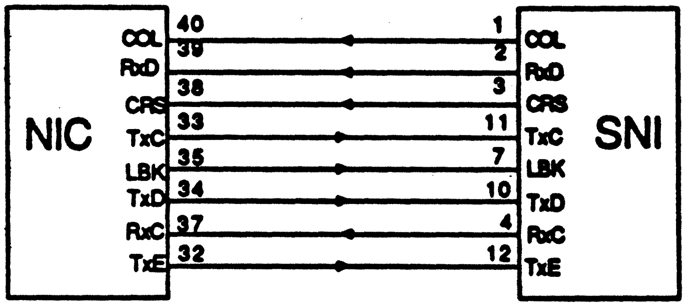


Figure 1 Network Interface

2. Processor Interface

The processor interface provides the ability to access the NIC's internal registers, receive interrupts from the NIC, and perform the hard reset operation.

2.1. Interrupt Generation

The NIC requests processor intervention by issuing an interrupt. The INT output of the NIC is active HIGH and should be connected to either an interrupt control unit, or directly to the processor. The INT output of the NIC goes low when the interrupt is reset by writing to the appropriate bit in the interrupt status register.

2.2. Hard Reset

All internal NIC registers and PLAs are reset when a hard reset is issued. When the /RST signal is activated for 8 DMA or network clocks (whichever is faster), a hard reset is accomplished. The /RST signal can be generated with either a power up RC circuit, or with an address decode to allow the software to accomplish the hard reset.

2.3. Register Access

The NIC is initialized and controlled through an array of registers, as a standard peripheral. All registers are 8 bits to maintain compatibility with 8 bit processors. The hardware designer must allow the processor to access these registers in regular Memory or I/O cycles. Register access cycles consist of two main parts, the register selection and the data transfer.

2.3.1. Register Selection

Four register address lines are provided (RA0-RA3) for selecting the proper register (16 registers are accessable at a time). These address lines can be latched with an address strobe provided by the processor and input to the ADS0 pin of the NIC, or if they are stable for the duration of the register access cycle, the ADS0 line can be pulled high to place the register selection in flow through mode. The latched mode using an address strobe is particularly useful when interfacing to a processor with a multiplexed address/data bus. The flow through mode is useful when interfacing to a processor with a demultiplexed address/data bus, or when register accesses and DMA operations occur on separate busses, as in the NIA demo board. In the latter case it is necessary to use a 8 bit transceiver (245) to isolate the data busses.

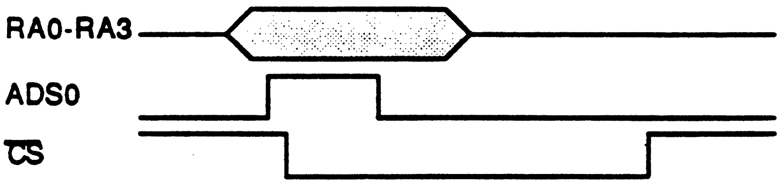


Figure 2 Register Selection - Latched

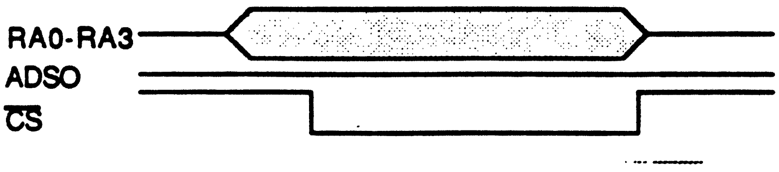


Figure 3 Register Selection - Flow Through

Once the register address is latched or stable, the NIC can be selected. The /CS input to the NIC is asserted to indicate that a NIC register is being accessed. This /CS signal would typically be a simple address decode.

2.3.2. Data Transfer

When /CS is active with the /SWR or /SRD signal, the processor must insert wait states until the /ACK signal is asserted. /CS may be asserted either before or after the assertion of /SRD or /SWR. The WAIT (shown below) must be generated and input to the processor. The acknowledge signal (/ACK) accomplishes arbitration for the NIC's internal data paths i.e. if a local DMA is in progress while a register is accessed, the register access will be deferred (by not asserting the /ACK signal) until the local DMA burst is complete. If a dual bus architecture is used, as in the DP839EB IBM PC demoboard (see AN-479 DP839EB Network Evaluation Board), the /ACK signal can be used directly to enable the 8 bit transceiver.

Once the /ACK signal is asserted the register access can proceed. During a read operation the NIC sources data on AD0-AD7 which is latched by the processor on the rising edge of the /SRD strobe. During a write operation, the NIC latches the data present on AD0-AD7 on the rising edge of the /SWR strobe.

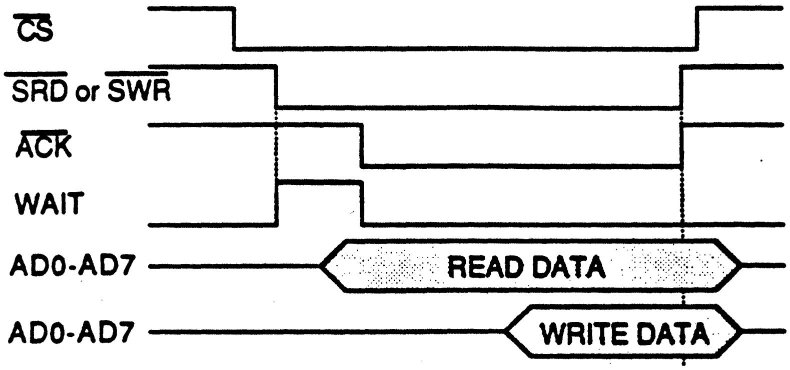


Figure 4 Register Data Transfer

Consult the DP8390 datasheet for the appropriate timing requirements.

3. Local DMA Interface

When receiving and transmitting packets, the NIC's local DMA channel is used to transfer data between memory and the NIC's internal 16 byte FIFO. At initialization, the Data Configuration Register is programmed with the proper configuration; 8 or 16 bit memory interface, byte ordering for 16 bit applications, and the appropriate FIFO thresholds and burst mode.

Local DMA transfers occur in bursts of an integral number of transfers depending on the selected FIFO threshold and burst mode. Once the FIFO threshold is reached, the NIC arbitrates for use of the bus, and when granted use of the bus, continues with the proper number of memory transfers.

3.1. Bus Arbitration

The bus request (BREQ) signal is asserted by the NIC to request use of the bus for the DMA transfer. External arbitration logic grants the NIC use of the bus by asserting the BACK signal. After the NIC sychronizes to the BACK signal, (typically four to eight DMA clocks) the memory transfers begin. The BACK signal should remain active until BREQ is deasserted.

3.2. Memory Transfer

The NIC provides multiplexed address/data (AD0-AD15) for use in the DMA operations. When 8 bit data paths are used, the lower 8 address bits must be latched, and when 16 bit data paths are used, all 16 address bits must be latched. The NIC's address strobe, ADS0, is used to latch the appropriate bits. ADS0 is only driven while the NIC is the master of the bus (BACK is asserted), at other times it is tri-stated. In single bus applications, it is possible to share the address latches with the processor (if the processor also gives multiplexed address/data signals) as long as the address strobe is handled properly. Many processors do not tri-state their address strobes when they are not the master of the bus, so it may be necessary to either tri-state their address strobe, or provide multiplexing logic for the address strobe (typically in a PAL).

Once the address is latched and present at the memory, the NIC asserts the /MRD or /MWR strobe, depending on whether a transmit or receive is in progress. Data must be stable at AD0-7(15) on the rising edge of the /MRD signal so the NIC can latch in the transmit data. The NIC presents data to the memory before the rising edge of the /MWR signal so the receive data can be written into memory.

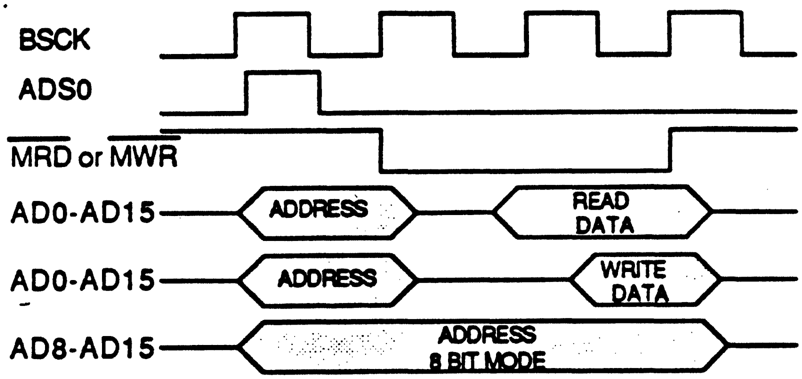


Figure 5 Local DMA cycle

3.3. DMA Burst Transfer Mode

The DP8390 DMA channel offers considerable flexibility when designing the local DMA interface. A separate DMA clock (BSCK) is used for the DMA cycles, allowing the DMA to easily be integrated into memory systems. In addition, the size of the DMA bursts is programmable. However, because of the finite size of the on chip FIFO, restrictions are placed on the local DMA interface.

There are two conditions which may cause the FIFO to overrun or underrun. The first condition occurs when the bus latency (the time between the assertion of BREQ and the granting of BACK) is longer than the time it takes to fill (or empty) the remaining locations in the FIFO. This places a restriction on the maximum allowable bus latency. The second condition occurs when the local DMA throughput is not greater than that of the network. The local DMA throughput is dependent on two factors, the DMA clock and the bus latency. This condition places further restrictions on the maximum allowable bus latency and the minimum DMA clock speed.

When the latency other devices on the bus can tolerate is crucial, as in systems using dynamic RAM that require refresh at defined intervals, the exact transfer mode should be used for the burst transfers. This mode guarantees that the NIC will only remain master of the bus for a fixed amount of time. In dual bus architectures where the NIC performs internal arbitration between register accesses and DMA operations, if a register access is attempted while a DMA operation is in progress, the processor must insert wait states until the DMA burst is complete.

On reception, no DMA transfers will occur unless the received Destination Address matches the node's Physical or Multicast address. This implies that the FIFO is used as a temporary storage for the received Destination Address. The BREQ signal is not asserted until 8 bytes have accumulated in the FIFO. BACK must be granted before the FIFO overruns. The FIFO will overrun if BACK is not granted within 3.6μsec for 2 and 4 byte FIFO thresholds and 2.8μsec for 8 byte

FIFO thresholds. These maximum allowable bus latencies (for 10Mbit/sec networks) assume that the empty/fill DMA transfer mode is used with at least a 7 MHz DMA clock is used. The Exact DMA transfer mode places further limitations on the allowable bus latencies for 2 and 4 byte thresholds.

When the exact burst DMA transfer mode is selected, the relationship between the DMA clock frequency, the burst size (the burst size is equal to the FIFO threshold when using the exact transfer burst mode) and the bus latency is more involved. The DMA must be able to transfer data into and out of the FIFO faster than the network over every time period equal to *8 * FIFO threshold (bytes) *period of the network clock (100ns for Ethernet).*

The graphs of figures 6 and 7 below show the range of usable FIFO thresholds, given the worst case bus latency that can be guaranteed to the NIC, and the frequency of the DMA clock (BSCK), to avoid FIFO underrun/overrun errors for byte and word mode transfers in Ethernet/Cheapernet applications. For a particular worst case bus latency, any DMA clock frequency to the right of the threshold curve is usable. If the system can not guarantee the NIC an appropriate bus latency, an architecture using a local buffer RAM should be considered.

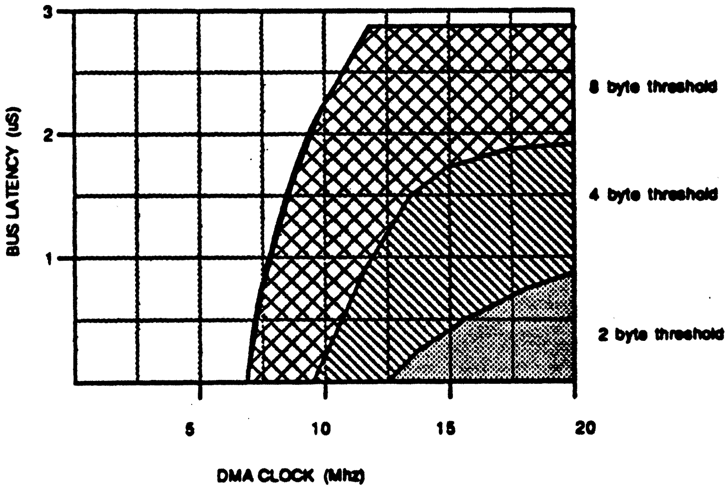


Figure 6 FIFO Threshold for Byte Mode

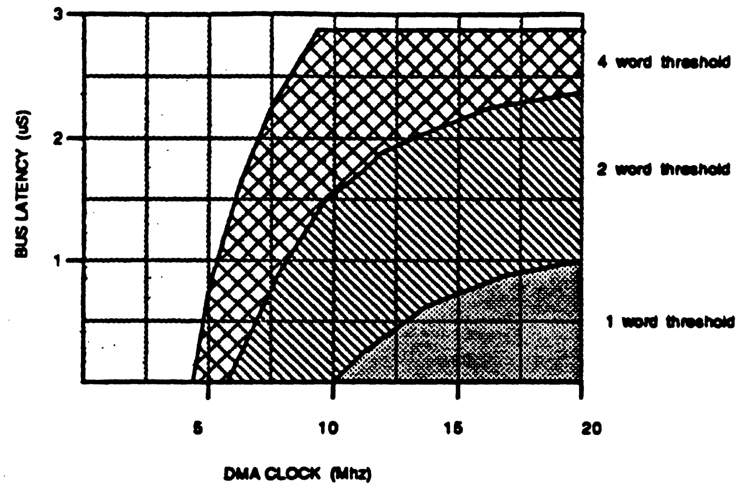


Figure 7 FIFO Threshold for Word Mode

Within the latency requirements of other devices on the bus, it is desirable to use bursts as long as possible to maximize the usable bandwidth for other devices on the bus. Since associated with each DMA burst is a DMA set up and recovery time (8-12 DMA clocks), if the bursts are longer, fewer bursts will occur, and less bandwidth will be used for DMA set up and recovery. In most applications, an eight byte threshold should prove to be a sufficient starting point, but in certain cases it may be advantageous to use shorter burst (when it is necessary to guarantee other devices certain bus latencies) or longer bursts (when it is desirable to minimize the NIC's utilization of the bus).

See the DP8390 Introductory Guide, Data Sheet and Update for further information. The empty/fill DMA burst mode is not supported on REV C of the DP8390 for normal operation (it is used during loopback diagnostics). Subsequent revisions of the DP8390 support this feature.

In STARLAN applications, where the network clock is 10 times slower, the bus latency criteria is much less rigid (60 μsec for 8 byte thresholds), and should not pose any design restrictions.

3.4. Wait State Insertion

With the mentioned bus interfacing features, it should be possible to design the DMA interface to avoid wait state insertion. However, if necessary, an integral number of wait states may be inserted into the DMA cycles as noted in the datasheet (timing section).

4. Remote DMA Interface

Remote DMA capabilities were included in the NIC to aid in the transfer of information between a local buffer RAM and the host system. The Remote DMA channel is particularly useful in applications that make the network interface appear as a standard I/O port, as in architectures similar to the NIA demoboard.

The Remote DMA capabilities of the NIC require minimal external components and provide a simple implementation. All of the control circuitry is provided by the DP8390. In addition, bus arbitration with the local DMA is accomplished within the NIC in such a way as to not lock out other devices on the bus.

4.1. Remote Read

During a remote read operation, data is read from the network memory and transferred to the port, whereupon it is read by either a system processor or DMA channnel. Each transfer follows the following protocol.

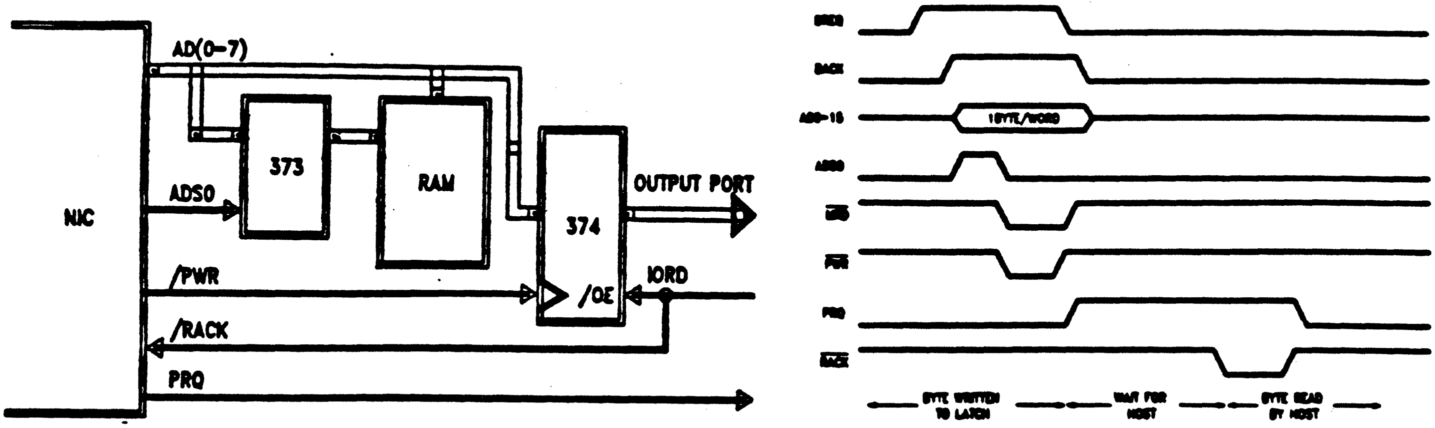


Figure 8 Remote Read Transfer

The Remote DMA channel arbitrates for the local bus by raising the BREQ signal. Upon receiving the BACK signal, the first half of the transfer begins. The 16 address lines are placed on the bus (AD0-15), and latched with ADSO into the resident 373. The /MRD signal is given to the memory and data flows to the 374. After two DMA clocks the /PWR signal latches the data into the 374, and the Remote DMA channel releases the bus. At this point the remote DMA byte count is decremented, the address is incremented and an interrupt is given to the system via the PRQ signal. Upon receiving the PRQ interrupt either the DMA controller or processor reads the data from the 374. The signal the system uses to read the 374 is used by the NIC as the read acknowledge, (RACK signal) to indicate the end a remote read transfer. Transfers continue until the remote DMA byte count reaches zero. It is only possible to read the remote DMA address during operation.

4.2. Remote Write

During a remote write operation, data is written into the network memory, once it is placed in the port. Each transfer follows the following protocol.

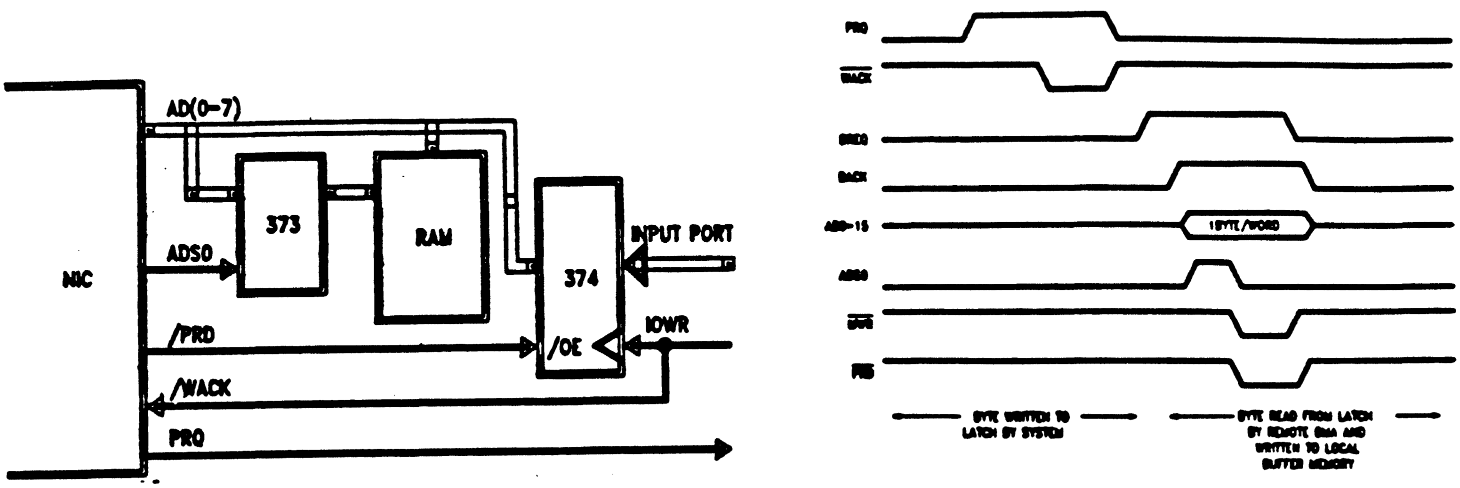

AD(0-7)
373
RAM
374
INPUT PORT
ADSO
NIC
/PRD
/OE
IOWR
/WACK
PRQ
PRQ
WACK
BREQ
BACK
AD0-15
1 BYTE/WORD
ADS0
PRD
MWR
BYTE WRITTEN TO LATCH BY SYSTEM
BYTE READ FROM LATCH BY REMOTE DMA AND WRITTEN TO LOCAL BUFFER MEMORY

Figure 9 Remote Write Transfer

The Remote DMA channel indicates that it is ready to receive a byte by sending an interrupt (PRQ) to the system. The PRQ signal remains high until the system writes to the port. The signal the system uses to write to the port is used by the NIC as the write acknowledge (WACK signal) to indicate that byte has been written to the 374. After the byte is in the latch, the Remote DMA channel arbitrates for the local bus as with the Remote Read. Once the address is latched and available at the memory, the output of the 374 is enabled with the /PRD signal and written into memory with the /MWR signal. At this point the remote DMA address is incremented and the byte count is decremented. Transfers continue until the remote DMA byte count reaches zero.

4.3. Higher Performance

The above byte transfer protocol, is equally applicable to word wide Remote DMA transfers. This modification doubles the transfer rate, but requires additional hardware. It is only possible to perform word wide Remote DMA transfers when word wide Local DMA transfers are utilized.

For even higher performance systems, it is possible to take advantage of burst mode DMA features of DMA controllers, by building a more sophisticated handshaking protocol with additional external logic. as possible, show single bus systems.

National
Semiconductor
Corporation

Guide to Loopback using the DP8390

August 26, 1986

Loopback capabilities are provided to allow certain tests to be performed to validate operation of the DP8390 NIC prior to transmitting and receiving packets on a live network. Typically these tests may be performed during power up of a node. The diagnostic provides support to verify the following:

1) Verify integrity of data path. Received data is checked against transmitted data.

2) Verify CRC logic's capability to generate good CRC on transmit, verify CRC on receive (good or bad CRC).

3) Verify that the Address Recognition Logic can

a) Recognize address match packets
b) Reject packets that fail to match an address

LOOPBACK MODES

Loopback modes are selected by programming the Transmit Configuration Register (TCR). Bits LB0 and LB1 select the type of loopback to be performed. The NIC supports three modes of loopback: Internal Loopback (figure 1), Loopback through an external encoder/decoder (figure 2) and Loopback through an external transceiver(figure 3).

LOOPBACK OPERATION IN THE NIC

Loopback is a modified form of transmission using only half of the FIFO. This places certain restrictions on the use of loopback testing. When loopback mode is selected in the TCR, the FIFO is split. A packet should be assembled in memory with programming of TPSR and TBCR0,TBCR1 registers. When the transmit command is issued the following operations occur:

Transmitter Actions

1) Data is transferred from memory by the DMA until the FIFO is filled. For each transfer TBCR0 and TBCR1 are decremented. (Subsequent burst transfers are initiated when the number of bytes in the FIFO drops below the programmed threshold.)

2) The NIC generates 56 bits of preamble followed by an 8 bit synch pattern.

3) Data transferred from FIFO to serializer.

4) If CRC=1 in TCR, no CRC calculated by NIC, the last byte transmitted is the last byte from the FIFO (Allows software CRC to be appended). If CRC=0, NIC calculates and appends four bytes of CRC.

5) At end of Transmission PTX bit set in ISR.

Receiver Actions

1) Wait for synch, all preamble stripped.

2) Store packet in FIFO, increment receive byte count for each incoming byte.

3) If CRC=0 in TCR, receiver checks incoming packet for CRC errors. If CRC=1 in TCR, receiver does not check CRC errors, CRC error bit always set in RSR (for address matching packets).

4) At end of receive, receive byte count written into FIFO, receive status register is updated. The PRX bit is typically set in the RSR even if the address does not match. If CRC errors are forced, the packet must match the address filters in order for the CRC error bit in the RS tc be set.

RESTRICTIONS USING LOOPBACK

Since the NIC is a half-duplex device, several compromises were required for the implementation of loopback diagnostics. Special attention should be paid to the restrictions placed on the use of loopback diagnostics.

1) The FIFO is split into two halves to allow some buffering of incoming data. The NIC transmits through one half of the FIFO and receives through the second half. Only the last five bytes of a packet can be examined in the FIFO, the DMA cannot store the loopback packet in memory. Thus loopback should be considered a modified form of transmission.

2) Splitting of the FIFO has some bus latency implications. The FIFO depth is halved, reducing the amount of allowed bus latency and thresholds of 8 and 12 bytes are not allowed. Also the Empty/Fill mode should be used to service the FIFO. In cases where the latency or Empty/Fill constraints cannot be accommodated, small packets can be transmitted. Also the FIFO must only be read when in Loopback mode, reading the FIFO in other modes will result in the NIC failing to issue the ACK signal properly.

3) The CRC logic is shared by the receiver and the transmitter. The NIC cannot generate and check CRC simultaneously. Thus software generated CRC must be used to check the receiver. The CRC bit in the TCR is provided to program the NIC to not append the CRC.

4) Address recognition logic must be checked indirectly through a small series of tests.

5) Between consecutive transmissions in loopback mode the TCR must be set to 00h then back to the desired loopback mode. This guarantees alignment of the FIFO pointers when data is read from the FIFO.

ALIGNMENT OF DATA IN THE FIFO

Data enters the receive half of the FIFO and is overwritten, thus only the last 8 bytes of a packet are retained. At the end of reception, the NIC writes three additional bytes into the FIFO. These are the lower and upper byte counts(two copies). These bytes overwrite those bytes previously received, thus a total of 5 received bytes can be examined in the FIFO. The length count found in the FIFO should agree with the number of bytes transmitted during loopback.

Since the FIFO uses a circulating write pointer, care should be taken to align data in the FIFO by transmitting L= N*8 +5 bytes (i.e. 13,21 etc). The resulting sequence of bytes read by consecutively reading the FIFO register is shown in figure 4. If CRC is appended the second through fifth byte will be the CRC appended by the NIC. This allows the CRC to be extracted from the NIC and compared to a previously calculated value for verification.

Before transmitting another loopback packet the TCR must be set to 00H to reset the FIFO pointers.

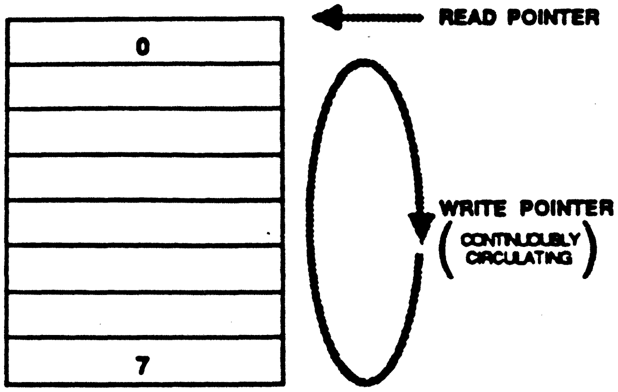


FIGURE 4

FIFO LOCATION	FIFO CONTENTS		
0	BYTE N-4	→	First Byte Read
1	BYTE N-3 (CRC1)	AR	Second Byte Read
2	BYTE N-2 (CRC2)		•
3	BYTE N-1 (CRC3)		•
4	BYTE N (CRC4)		•
5	LOWER BYTE COUNT		•
6	UPPER BYTE COUNT	→	Last Byte Read
7	UPPER BYTE COUNT		

"ALIGNMENT OF PACKET IN FIFO FOLLOWING LOOPBACK"

RESTRICTIONS USING WORD WIDE TRANSFERS

Since the FIFO is split, only half cf each word is transferred into the transmit portion of the FIFO. Thc BOS bit in the DCR can be used to select which half of the word is written into the FIFO. (See figure 5.)

> **NOTE:**
>
> Even though a word is transferred to the NIC, only a byte is transmitted in the loopback packet. To properly transfer all the bytes in the loopback packet, the byte count must be 2X the number of bytes assembled in the loopback packet.

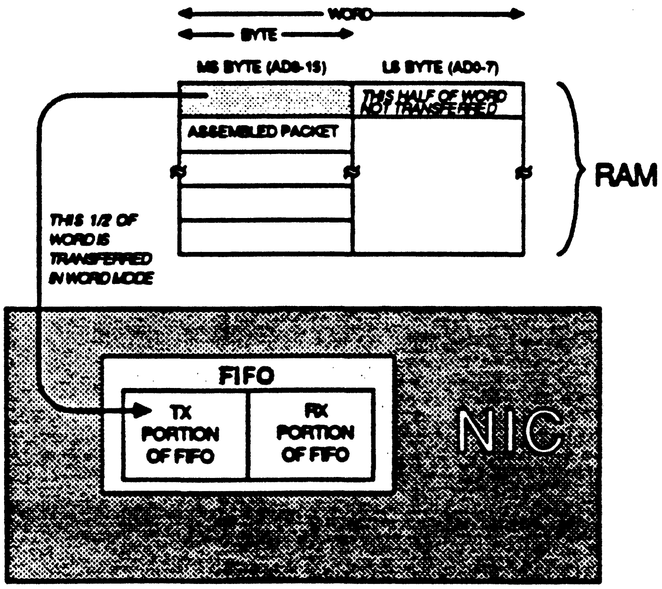


FIGURE 5

EXAMPLES

The following examples show what results can be expected from a properly operating NIC during loopback. The restrictions and results of each type of loopback are listed for reference. The loopback tests are divided into two sets of tests. One to verify the data path, CRC generation and byte count through all three paths. The second set of tests uses internal loopback to verify the receiver's CRC checking and address recognition. For all of the tests the DCR was programmed to 40h.

PATH	TCR	RCR	TSR	RSR	ISR
NIC Internal	02	00	53(1)	02(2)	02(3)

Notes

(1) Since carrier sense and collision detect inputs are blocked during internal loopback, carrier and CD heartbeat are not seen and the CRS and CDH bits are set.

(2) CRC errors are always indicated by receiver if CRC is appended by the transmitter.

(3) Only the PTX bit in the ISR is set, the PRX bit is only set if status is written to memory. In loopback this action does not occur and the PRX bit remains 0 for all loopback modes.

PATH	TCR	RCR	TSR	RSR	ISR
NIC External	04	00	43(1)	02	02

Notes

(1) CDH is set, CRS is not set since it is generated by the external encoder/decoder.

PATH	TCR	RCR	TSR	RSR	ISR
NIC external	06	00	03(1)	02	02(2)

Notes

(1) CDH and CRS should not be set. The TSR however, could also contain 01,03,07 and a variety of other values depending on whether collisions were encountered or the packet was deferred.

(2) Will contain 08 if packet is not transmittable.

General: During external loopback the NIC is now exposed to network traffic, it is therefore possible for the contents of both the Receive portion of the FIFO and the RSR to be corrupted by any other packet on the network. Thus in a live network the contents of the FIFO and RSR should not be depended on. The NIC will still abide by the standard CSMA/CD protocol in external loopback mode. (i.e. The network will not be disturbed by the loopback packet).

CRC and ADDRESS RECOGNITION

The next three tests exercise the address recognition logic and CRC. These tests should be performed using internal loopback only so that the NIC is isolated from interference from the network. These tests also require the capability to generate CRC in software.

The address recognition logic cannot be directly tested. The CRC and FAE bits in the RSR are only set if the address of the packet matches the address filters. If errors are expected to be set and they are not set, the packet has been rejected on the basis of an address mismatch. The following sequence of packets will test the address recognition logic. The DCR should be set to 40, the TCR should be set to 03 with a software generated CRC.

PACKET CONTENTS			RESULTS
TEST	ADDRESS	CRC	RSR
Test A	Matching	Good	01(1)
Test B	Matching	Bad	02(2)
Test C	Non-Matching	Bad	01

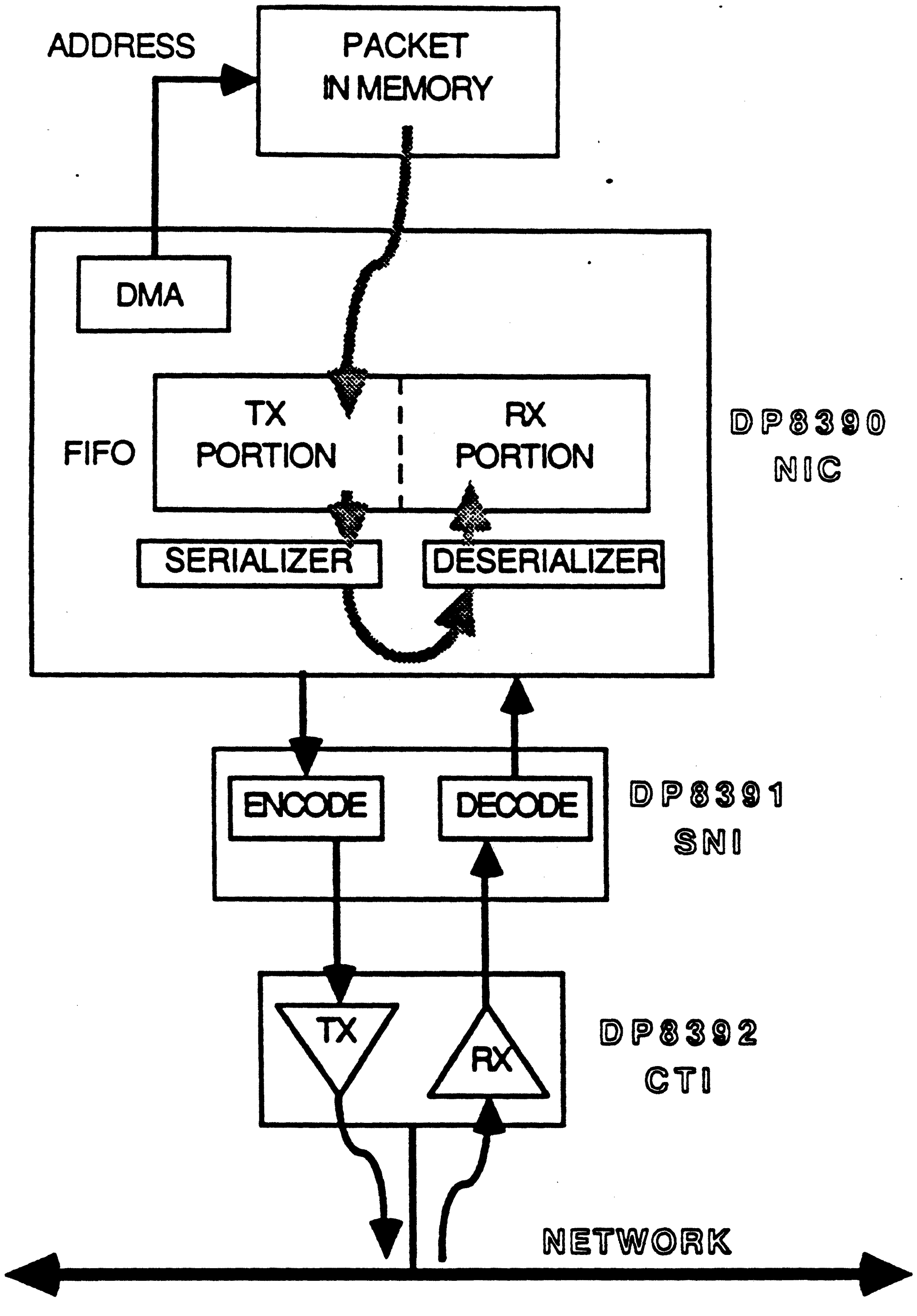
Notes

(1) Status will read 21 if multicast address used

(2) Status will read 22 if multicast address used

General: In test A, the RSR is set up. In test B the address is found to match since the CRC is flagged as bad. Test C proves that the address recognition logic can distinguish a bad address and does not notify the RSR of the bad CRC. The receiving CRC is proven to work in test A and test B.

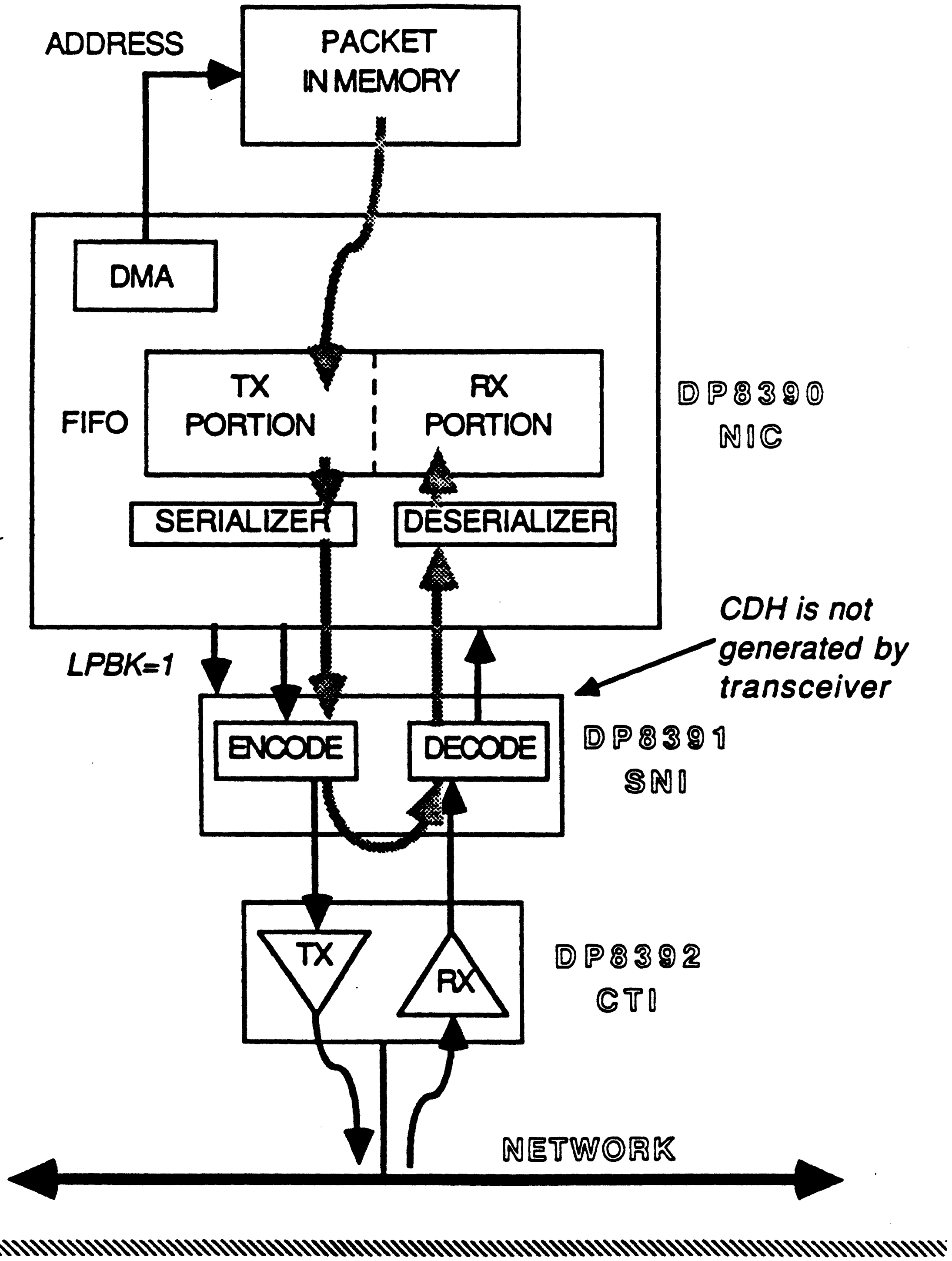
A series of flowcharts is provided for an example set of tests for loopback.



MODE 1

Things to expect: COL,CRS inputs blocked, TSR will indicate error.
TXD, TXE disabled.

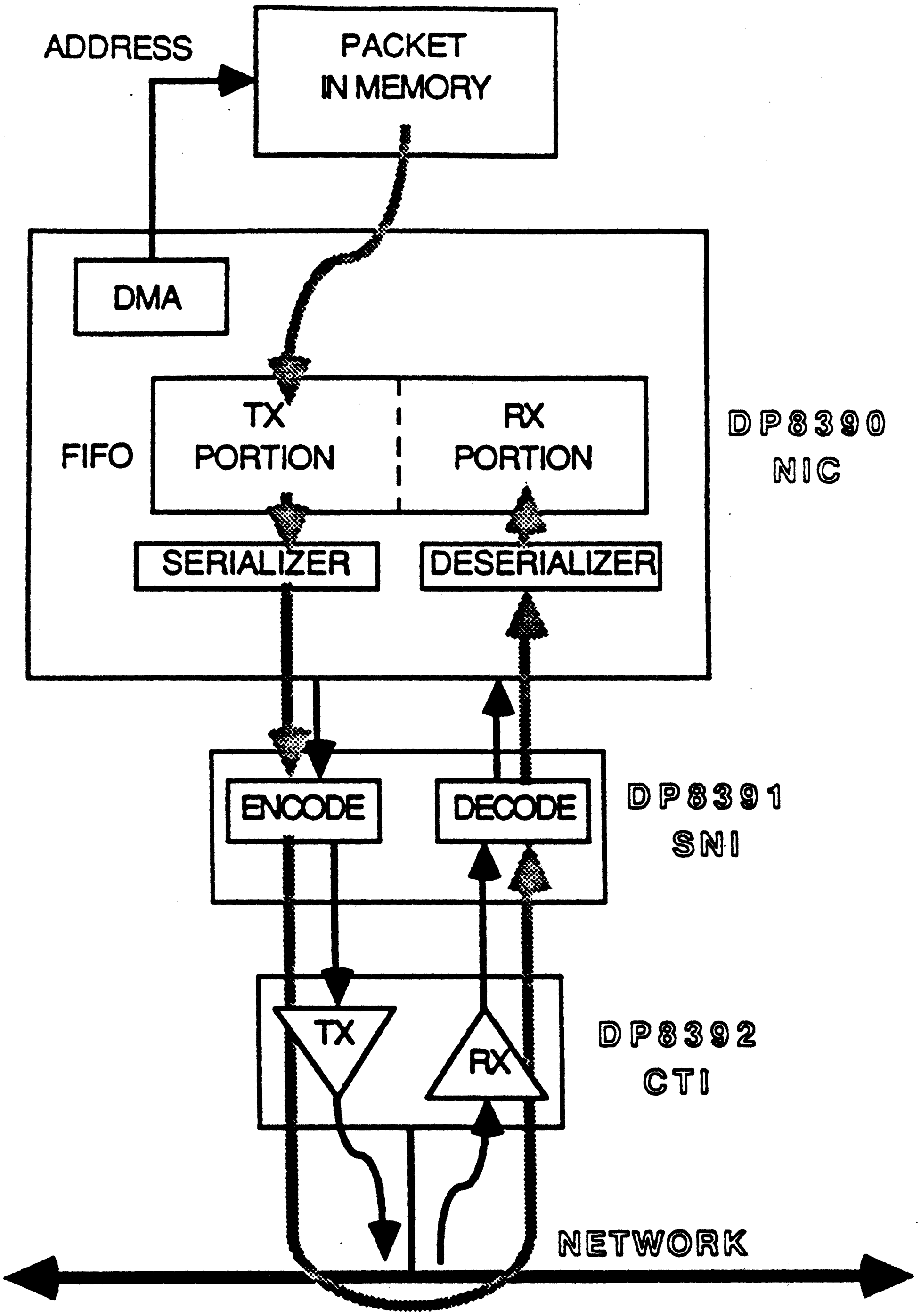
FIGURE 1



MODE 2

NOTES: LPBK output becomes active, CDH will be set in TSR since it is not generated by the SNI

FIGURE 2

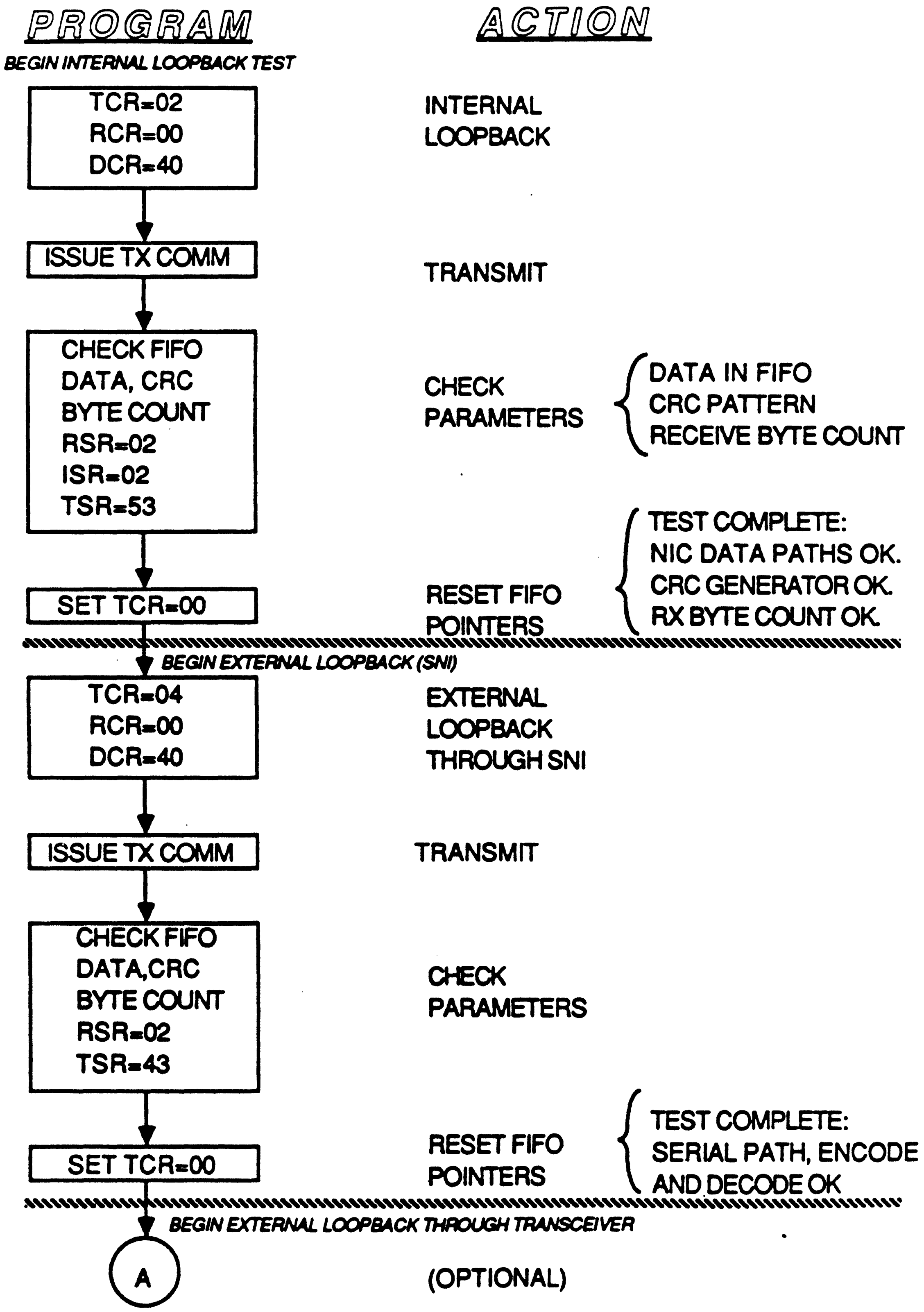


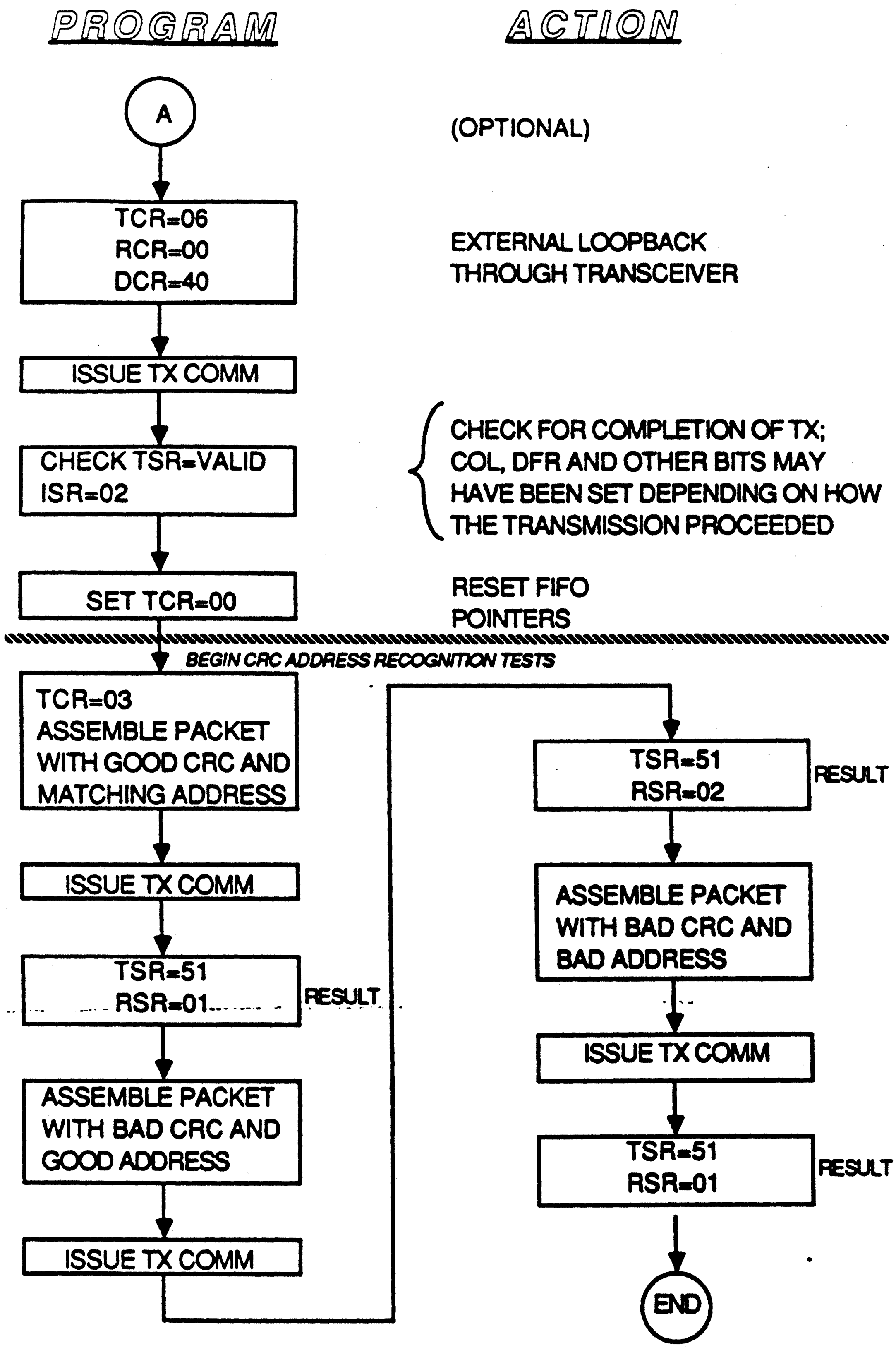
MODE 3

NOTES: Loopback success is dependent on being able to Tx on network. Rx data in FIFO Rx Status can be destroyed by any subsequent packet.

Recommend assisted loopback

FIGURE 3


PROGRAM
ACTION
BEGIN INTERNAL LOOPBACK TEST
TCR=02
RCR=00
DCR=40
INTERNAL
LOOPBACK
ISSUE TX COMM
TRANSMIT
CHECK FIFO
DATA, CRC
BYTE COUNT
RSR=02
ISR=02
TSR=53
CHECK
PARAMETERS
DATA IN FIFO
CRC PATTERN
RECEIVE BYTE COUNT
TEST COMPLETE:
NIC DATA PATHS OK.
CRC GENERATOR OK.
RX BYTE COUNT OK.
SET TCR=00
RESET FIFO
POINTERS
BEGIN EXTERNAL LOOPBACK (SNI)
TCR=04
RCR=00
DCR=40
EXTERNAL
LOOPBACK
THROUGH SNI
ISSUE TX COMM
TRANSMIT
CHECK FIFO
DATA,CRC
BYTE COUNT
RSR=02
TSR=43
CHECK
PARAMETERS
SET TCR=00
RESET FIFO
POINTERS
TEST COMPLETE:
SERIAL PATH, ENCODE
AND DECODE OK
BEGIN EXTERNAL LOOPBACK THROUGH TRANSCEIVER
A
(OPTIONAL)

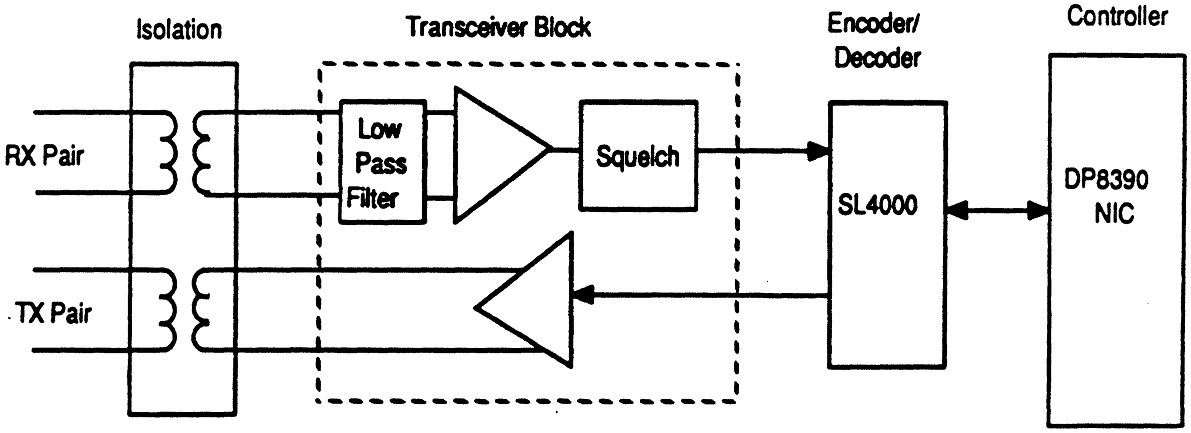

PROGRAM
ACTION
A
(OPTIONAL)
TCR=06
RCR=00
DCR=40
EXTERNAL LOOPBACK THROUGH TRANSCEIVER
ISSUE TX COMM
CHECK TSR=VALID
ISR=02
CHECK FOR COMPLETION OF TX; COL, DFR AND OTHER BITS MAY HAVE BEEN SET DEPENDING ON HOW THE TRANSMISSION PROCEEDED
SET TCR=00
RESET FIFO POINTERS
BEGIN CRC ADDRESS RECOGNITION TESTS
TCR=03
ASSEMBLE PACKET WITH GOOD CRC AND MATCHING ADDRESS
ISSUE TX COMM
TSR=51
RSR=01
RESULT
ASSEMBLE PACKET WITH BAD CRC AND GOOD ADDRESS
ISSUE TX COMM
TSR=51
RSR=02
RESULT
ASSEMBLE PACKET WITH BAD CRC AND BAD ADDRESS
ISSUE TX COMM
TSR=51
RSR=01
RESULT
END

NATIONAL
SEMICONDUCTOR
CORPORATION

StarLAN with the DP839EB Evaluation Board

Overview

Because of the identical packet structures and protocols between StarLAN (1base5) and Ethernet, the DP8390 Network Interface Controller (NIC) can operate in both systems. To evaluate the DP8390 in StarLAN applications, the DP839EB Evaluation Board can be used with a "daughter card" that replaces the Ethernet/Cheapernet front end with a StarLAN front end. The StarLAN front end consists of an RS-422 type transceiver and a 1Mb/s Manchester endcoder/decoder, as shown below. The SL4000, manufactured by Semicustom Logic, is used to perform the required ENDEC functions, and provides a direct interface to the DP8390. Further information on the SL4000 is available in the SL4000 datasheet.



StarLAN Frontend for the DP8390

Cabling

A significant number of StarLAN networks are expected to use existing twisted pair telephone wiring. In this type of environment, the DTEs will be connected to wall outlets, which in turn, will be connected to wiring closets where the hubs will be located. The cabling used typically will consist of 26 - 22 gauge, unshielded twisted pairs and maximum cable length will be approximately 250 meters (800ft).

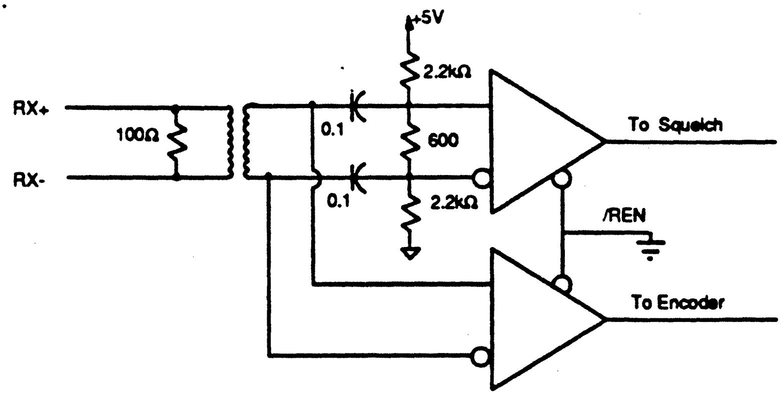
Transceiver

The transceiver connects to two twisted pair phone wires, one for transmit, the other for receive. Similar to Ethernet, the cabling is isolated by two pulse transformers. Some pulse transformers also provide rise time limiting to reduce EMI. The transceiver circuitry is based on the DS8923 dual receiver/driver combination. Two of the receivers are used to provide receive and squelch functions.

Squelch

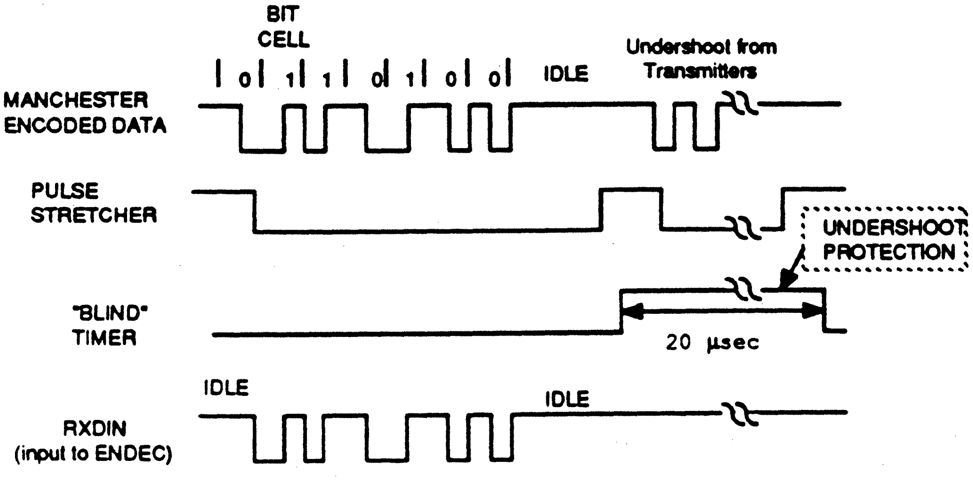
Since the cabling may be bundled together and routed close to heavy electrical equipment, squelch circuitry is necessary to reject signals generated from crosstalk between adjacent wires and impulse noise from large equipment. Proper noise immunity may be implemented using a second-order Butterworth filter with a 2MHz cutoff and setting a 600 mV squelch level. Because RS-422 receivers typically have 200mV threshold level, these inputs must be skewed to 600mV. This may be implemented by using a resistor ladder which holds the inputs 600mV apart (as shown below). When an incoming signal exceeds the 600mV threshold, the receiver is enabled.

The squelch circuitry must also reject signals of proper amplitude but with improper timing. Valid Manchester encoded data has transitions, at the longest, 1 usec apart. This squelch circuitry may be implemented by using a pulse stretcher which counts the time between sucessive transitions in the Manchester encoded data. If the transitions occur less than 1.6usec apart , the pulse stretcher retriggers and the data is declared valid; otherwise data to the encoder (SL4000) is squelched out.



Squelch Level Adjustment

At the end of transmission, the receivers are vulnerable to noise generated from the transmitters shutting off from IDLE. During this period, undershoot may be produced from the transmitter for a duration up to 20usec. This undershoot may falsely interpreted by the decoder as valid; thus, the squelch circuitry must "blind" the receiver during this period. Both the pulse stretcher and "blind" timer are implemented in a 16R6 PAL. The state diagram and PAL listing are shown at the end of this document.



Squelch Timing

Transmitter

The transmitter is comprised of one RS-422 driver provided in the DS8923 package. The Driver is enabled using the /CTS output of the SL4000. /CTS is asserted coincident with the first bit of valid data and is deasserted two bit time following the last bit. This allows generation of the 2 bit idle signal marking the end of the packet. When /CTS is deasserted, the RS-422 driver is disabled.

SL4000 Interface to the NIC

The SL4000, from Semi Custom Logic, interfaces directly to the serial interface pins of the DP8390 NIC. The PSEL pin of the SL4000 (pin 19) must be tied LOW to select the proper polarty for these signals (active HIGH). For more infomation, consult the SL4000 data sheet. An additional /RESET line is required by the SL4000 on power up. This can be connected to an RC network or to /RST (pin 41) of the DP8390 NIC.

82C550A Interface to the NIC

The 82C550A, from Chips & Technologies, is similar in function to the SL4000, but has on-chip squelch; hence the squelch circuitry discussed above is not necessary. The 82C550A interfaces to the DP8390 via 5 inverters to provide the proper polarity. The normal mode is selected since the input_threshold level (600mV) needs to be adjusted externally. For more information consult the 82C550A datasheet.

Building A StarLAN Daughter Card for the NIA

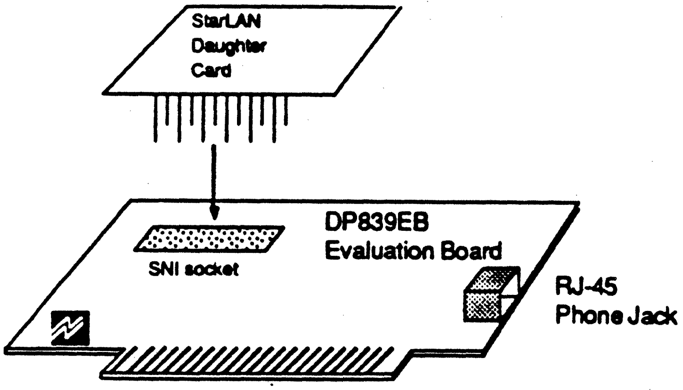
The DP8391 Serial Network Interface of the DP839EB Evaulation Board has been socketed to allow insertion of a StarLAN daughter card in its place. Unused pins on the DP8391 have been wired with additional signals that are necessary for a StarLAN daughter card. The phone jack is connected to the receiver and transmit pairs. The Schematic of a working daughtercard is attached.

SNI Socket (U9) pin	Connection
1	COL
2	RXD
3	CRS
4	RXC
6	GND
7	LBPK
10	TXD
11	TXC
12	TXE
13	TX- (pin 2 of phone jack)
14	TX+ (pin 1 of phone jack)
16	/RST
19	+5V
21	RX- (pin 6 of phone jack)
22	RX+ (pin 3 of phonejack)

Daughter Card Pin Assignment

Installation of the Daughter Card

Once the daughter card has been assembled, the DP8391 Serial Network Interface chip (socketed) can be removed and replaced with the daughter card. Prior to installing the daughter card, the following jumpers must be removed: J1C - J7C, J1E - J7E, and JY. All demo software that is provided with the DP839EB works in StarLAN applications as well. The DP839EB may be attached to the StarLAN network by connecting the 8 pin RJ-45 modular jack to the hub via twisted pair phone cable.



StarLAN Daughter Card Installation

Supporting Documents

The following references can be used to obtain further information .

- Advanced Peripherals IEEE 802.3 Local Area Network Guide
- DP8390 REV C update, December 10, 1986
- Memo: DP839EB Upgrade to REV C DP8390, December 15, 1986
- IEEE 802.3 1Base5 ("StarLAN"), draft H, December 1986
- SL4000 Data Sheet (a product of Semi-Custom Logic)
- 82C550A Data Sheet (a product of Chips and Technology)

If you have any further questions regarding StarLAN, contact Wesley Lee at (408) 721-3221.

Considerations For Using Rev. C Silicon

(1) There is a maximum ratio between the DP8390 network and DMA clocks. In order for the 4 byte packet header to be properly written by the DP8390, the DMA clock to Network clock may not be greater than 4:1; thus, in StarLAN applications, the DMA clock may not be greater than 4MHz. This problem, however, can be circumvented by manipulating the packet header under software control. If you are using a DMA clock which is greater than the 4:1 ratio, the DP8390 occassionaly copies the Lower Byte Count into the header twice, and fails to write the Upper Byte Count. The Upper byte count, however, is still available, although somewhat disguised. By subtracting the Next Page Pointer (second byte in the header) with the Next Page Pointer of the previous packet, the Upper Byte Count can be calculated.

Due to the architecture of the DP8390, this "restriction" will exist in all remaining revisions.

(2) Due to the asynchronous nature between the local and remote DMAs, a race condition exists which may cause the local DMA to use the remote DMA's address counter or visa versa. This problem has been indentifed and it was found that the problem is fixed using a DMA clock synchronous to the transmit clock of the encoder (SL4000), or a clock derived from the transmit clock. As shown in the schematic, an HC74 flip-flop is used to divide down the 8MHz clock of the SL4000 to 4MHz, and drive the DMA clock of the DP8390.

This problem will be fixed in revision D.

(3) The "send packet" command has been found to be non-functional under heavy network conditions for StarLAN applications; hence, it is not recommend that this command be used. Instead, used the Remote Read DMA and update BNDRY under software control. Note that there are special conderations for updating BNDRY as specified in section 2.1 of the DP8390 REV C UPDATE (December 10, 1985). To reiterate briefly, BNDRY must always be kept at least one 256 byte buffer behind the CURR pointer.

Squelch PAL Listing
(ABEL Format)

```
module starlan_sqlch flag '-r2',t4'     title 'StarLan Squelch
Wesley Lee        National Semiconductor  10/4/86 '
sqlchdevice'p16R6';

"Define Pins

clk,datain,sqlch2in,data2in,enb
            pin 1,2,3,4,11;

dataout,q0,q1,q2,q3,dd,sqlchout,dd2
            pin 12,13,14,15,16,17,19,18;

ck,x,z = .C., .X., .Z.;                    "clock, don't care, high imp.


"Squelch States

          s1 = ^b0001;
          s3 = ^b0011;
          s7 = ^b0111;
          s6 = ^b0110;
          s12 = ^b1100;
          s4 = ^b0100;
          s5 = ^b0101;
          s13= ^b1101;
          s9 = ^b1001;
          s8 = ^b1000;
          s10 = ^b1010;
          s15 = ^b1111;
          s0 = ^b1110;
          s11= ^b1011;
```

```
            s2 = ^b0010;
            s14 = ^b0000;




state_diagram [q0,q1,q2,q3]

state s15:
            if !dd & sqlch2in then1
                        else 15;
state s1:
            if !dd then 14
                        else 2;
state s2:
            if !dd then 14
                        else 3;
state s3:
            if !dd then 14
                        else 4;state s4:
state s4:
            if !dd then 14
                        else 5;
state s5:
            if !dd then 14
                        else 6;
state s6:
            if !dd then 14
                        else 7;
state s7:
            if !dd then 14
                        else 8;
state s8:
            if !dd then 14
                        else 9;
state s9:
            if !dd then 14
                        else 10;
state s10:
            if !dd then 14
                        else 11;

state s11:
            if !dd then 14
                        else 15;
state s12:
            goto 1;
state s13:
            goto 1;
state s14:
            goto 1;
state s0:
            goto 1;
```

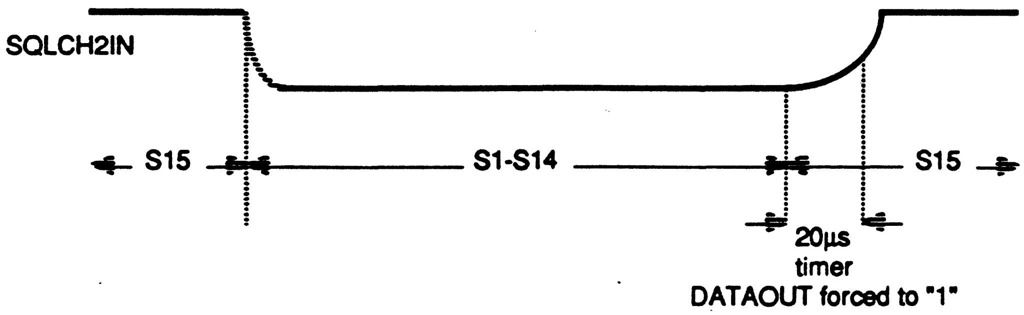
Equations

```
        dd := datain;
        dd2 := dd;
        sqlchout = q0 & q1 & q2 & q3;
        dataout = q0 & q1 & q2 & q3 # data2in;
```

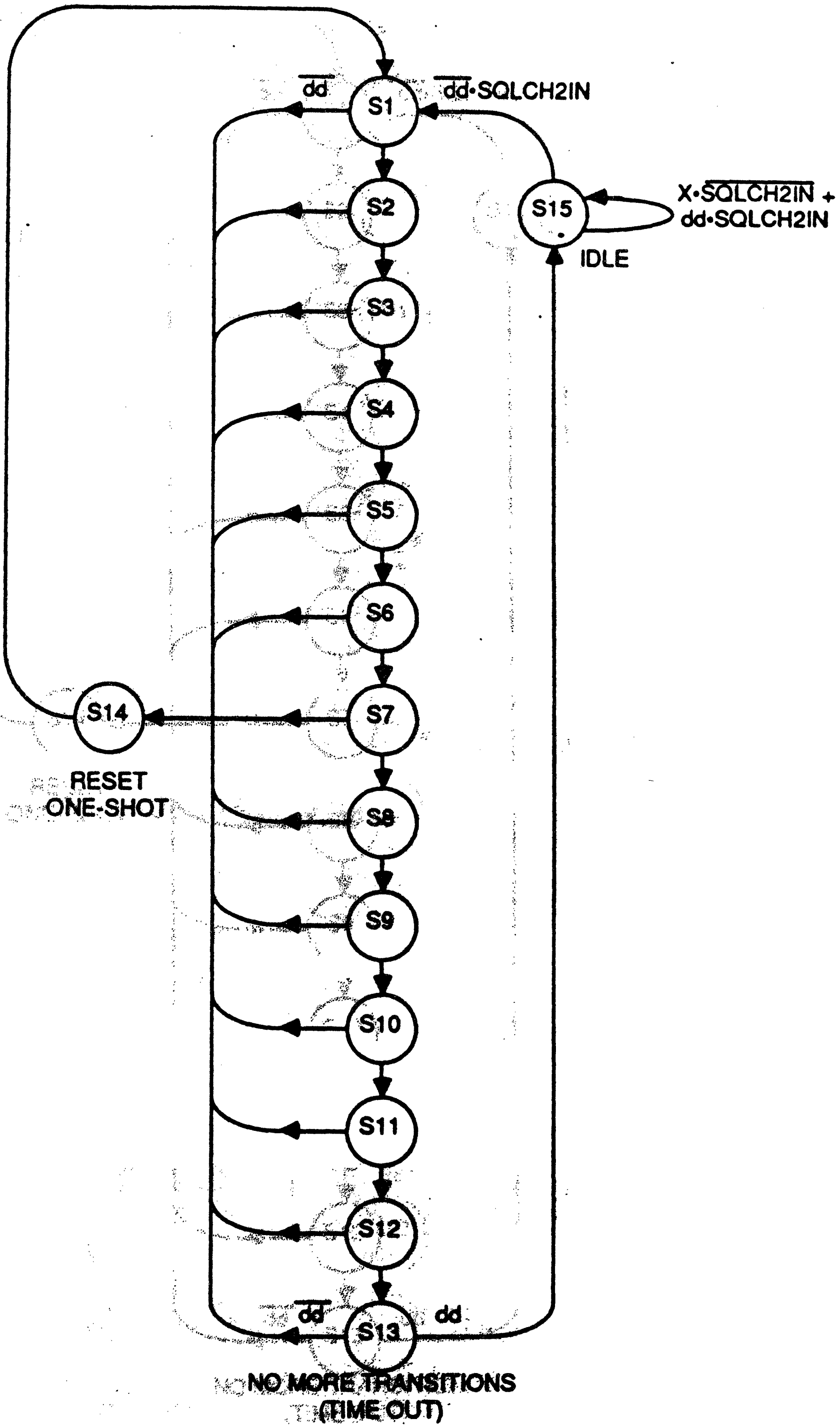
"Description:

" This PAL implements the squelch functions for StarLan applications using the SL4000 encoder/decoder. "Incoming "data (datain) is first synchronized with the 8MHz clock; then sampled for the first 'zero' in the "data stream. The squelch "timer then begins on the next 'one' in the data stream and is reset upon each "'zero' in the data. The squelch eventually "times out when the input data stays high for greater than 1.6us. "At the end of the timeout, an RC time constant "prevents data from being received for another 20us.

end starlan_sqlch



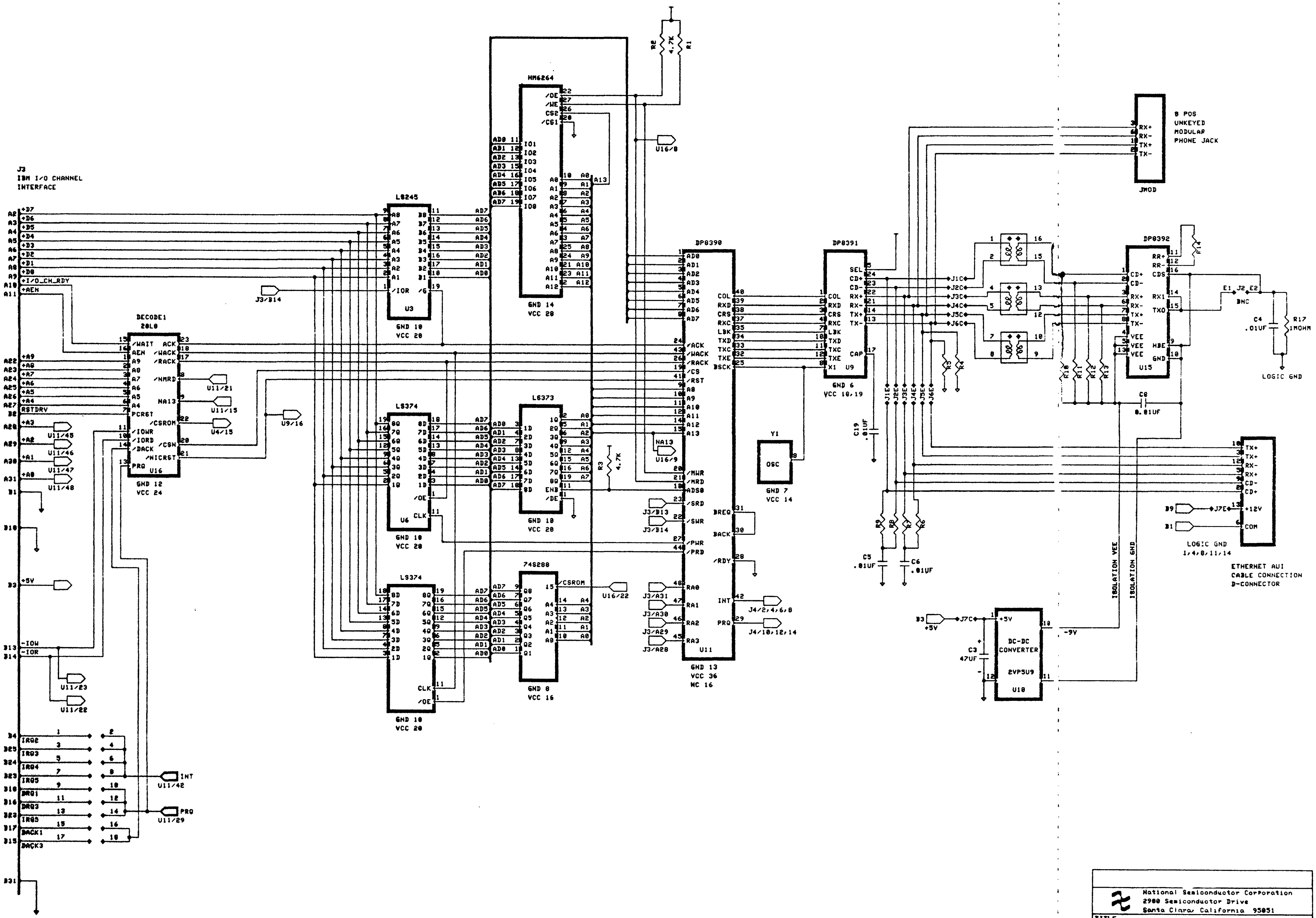
Squelch PAL Timing



dd = Synchronized Data Input

Squelch State Machine

REVISIONS				
REV.	SHEET	DESCRIPTION	DATE	APPROVED
C	1	IBM PC/AT ETHERNET/CHEAPERNET/STARLAN	2/10/87	



National Semiconductor Corporation
2900 Semiconductor Drive
Santa Clara, California 95051

TITLE	
REV D BOARD SUPPORTS STRING I/O AND DMA	
ENGINEER DATE DOUG WILSON 408-721-3427	DESIGN SITE

T4T34 /VCS/USER/ERICK REVD.BRD

```
PAL20L8                                             National Semiconductor
I/O Address Decode Plus for NIA                     March 2, 1987
DECODE1                                             Doug Wilson

A9   A8    A7   A6   A5    A4  PCRST  /NMRD  NA13 /IORD  /IOWR GND
PRQ /DACK /WAIT AEN /RACK /WACK /CSX /CSN   /NICRST  /CSROM /ACK  VCC

CSN =  /AEN* A9* A8* /A7* /A6* /A5* /A4* IOWR +
       /AEN* A9* A8* /A7* /A6* /A5* /A4* IORD

NICRST = PCRST

RACK = /AEN* A9* A8* /A7* /A6* /A5* A4* PRQ* IORD +
        DACK*  IORD

WACK = /AEN* A9* A8* /A7* /A6* /A5* A4* PRQ* IOWR +
        DACK*  IOWR

CSX =   CSN* IORD +
        CSN* IOWR +
       /AEN* A9* A8* /A7* /A6* /A5* A4* IORD +
       /AEN* A9* A8* /A7* /A6* /A5* A4* IOWR

IF (CSX)
WAIT = /ACK *  CSN   +
       /PRQ * /CSN

CSROM = /NA13 * NMRD
```

Description

This pal performs the I/O decodes for selecting the NIC, and the handshake signals for NIC's remote dma. The pal supports the dma channels of the PC for remote DMA transfers with the NIC and also allows the use of string I/O between 80286 PC's and NIC's remote DMA.

Using DECODE1 fixes the I/O BASE of the card at 300h. NIC registers fall in the space 300h - 30fh. To use the string I/O port, reads and writes are done to port 310h.

Wait states are inserted (WAIT) to the PC bus when register accesses are given and the NIC is busy performing other operations (such as local bursts). When the NIC is ready, /ACK is given and no (more) wait states are inserted.

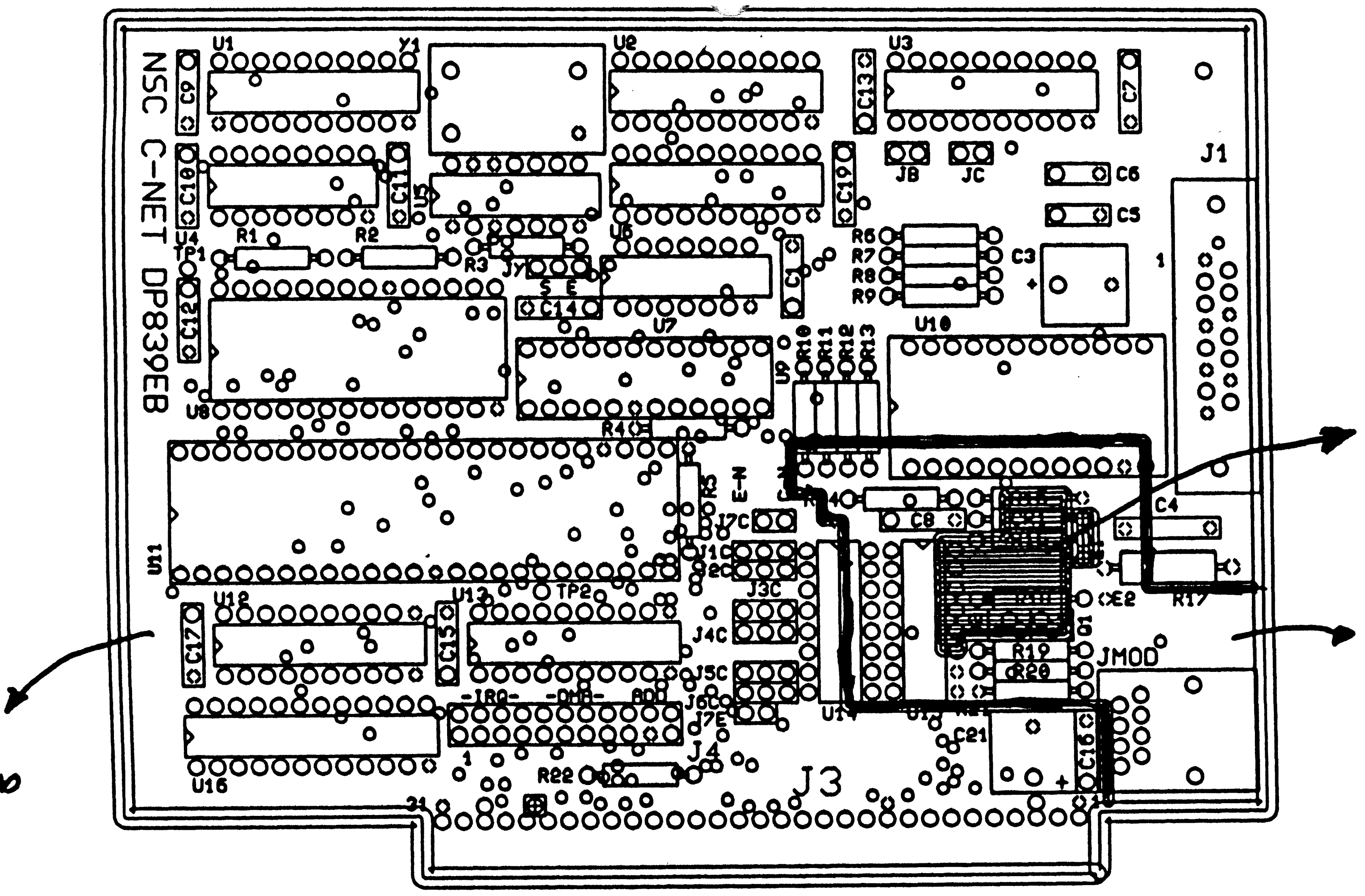
Wait states may also be inserted during remote DMA operations and 80286 machines using string I/O's. WAIT occurs during a remote read if the PC AT's /IORD goes low before the DP8390's PRQ goes high. Similarly, WAIT occurs during a remote write if the PC AT's /IOWR goes low before the NIC's PRQ goes high.

NIC registers are accessed when CSN (Chip Select NIC) is asserted. The IORD and IOWR terms are included to ensure that the address lines are valid when CSN is given.

The RACK and WACK signals are used by the NIC's remote DMA channel to acknowledge the end of a single read or write operation through the remote DMA I/O ports. These port are addressable by the PC DMA channel with DACK and IORD or IOWR, or by addressing the I/O location 310h (with string I/O's).

CSX is used to enable the TRI-STATE output of WAIT during a register access (CSN), and during string I/O to the remote DMA's I/O port (CSX).

CSROM provides address decode for the address PROM. The card's unique Ethernet address is transferred to the system using the NIC's remote DMA.



NO PLANE AT ALL

LOGIC GROUND

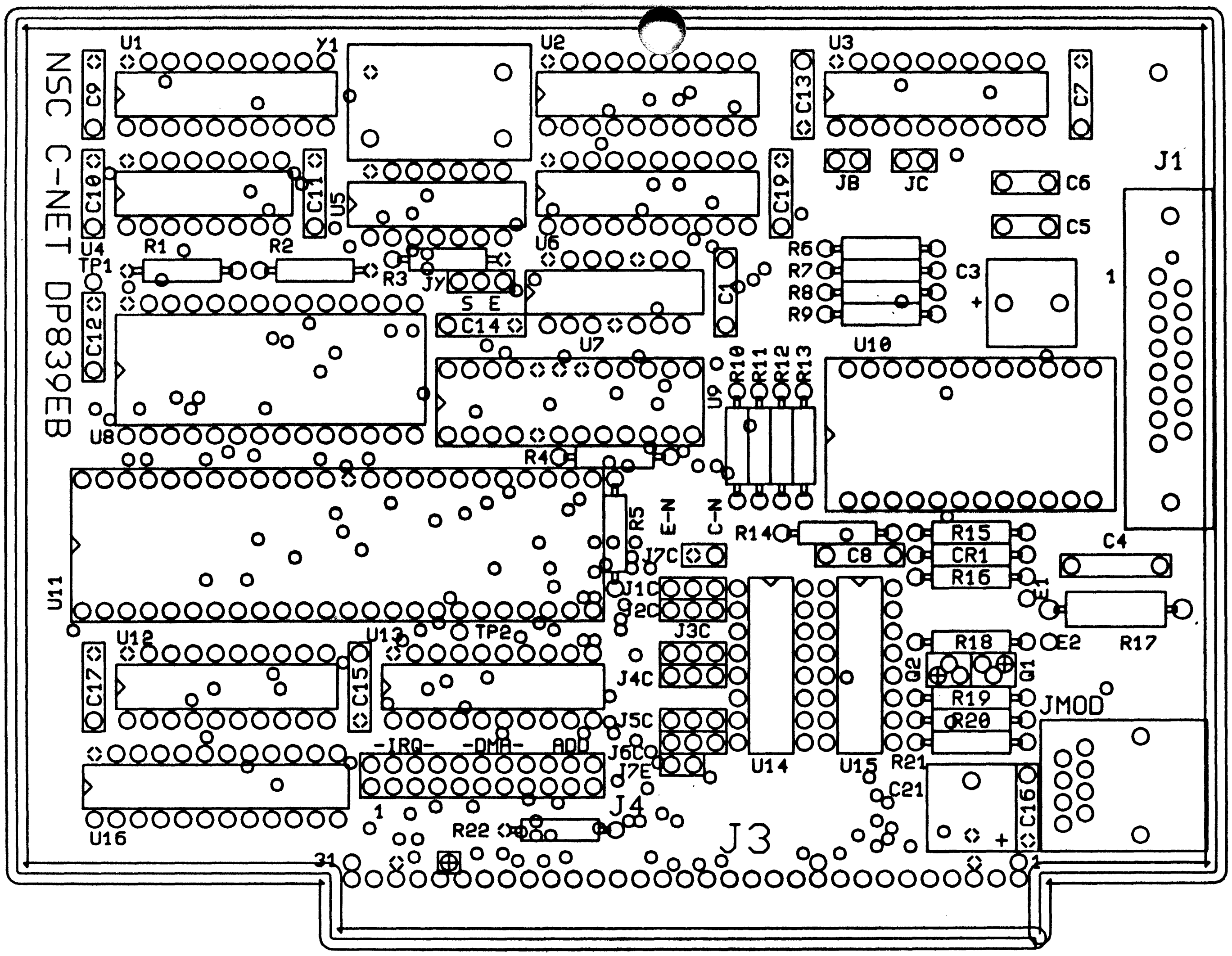
ISOLATION GROUND

NATIONAL
C-NET BOARD
551 A 201-01
GROUND PLANE
N C I NA01B 4-24-86

JH1

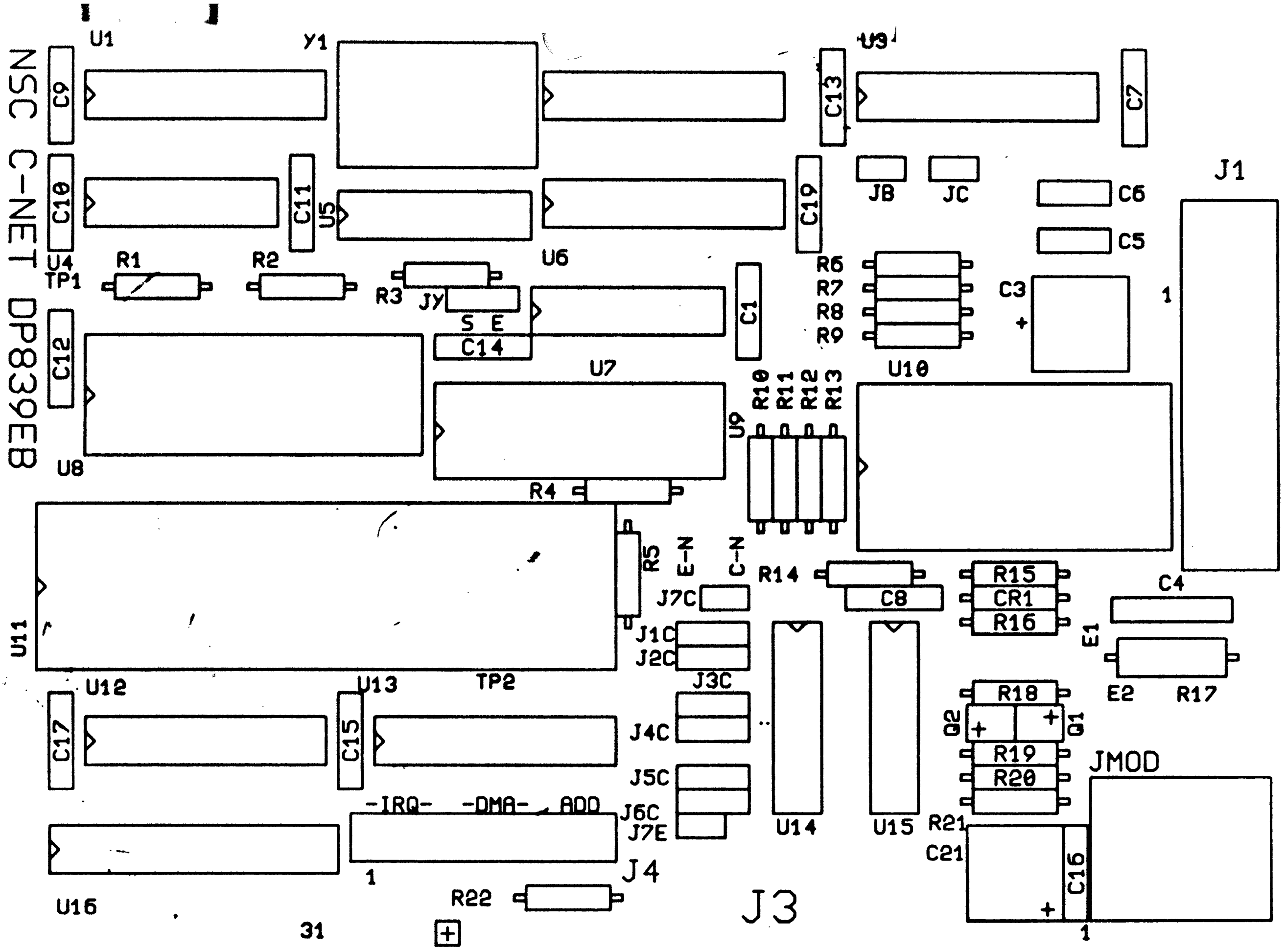
★ THERE SHOULD BE NO MORE THAN 1½ INCHES BETWEEN THE DP8392 CTI AND THE END OF THE BNC CONNECTOR

★ PC BOARD MOUNT BNC CONNECTORS DO NOT MEET THE IEEE 802.3 CAPACITANCE SPEC – USE BRACKET MOUNT BNC CONNECTORS



TH1

NATIONAL
C-NET BOARD
551 A 201-01
VOLTAGE PLANE
N C I NA01B 4-24-86



TH1

NATIONAL
C-NET BOARD
551 A 201-01
COMPONENT SIDE SILKSCREEN
N C I NA01B 4-24-86

Rev D Board Parts List 1/27/87

PARTS LIST

Part No.	Description	Quantity
U1	74LS373N	1
U2,U6	74LS374N	2
U3	74LS245N	1
U4	74S288	1
U8	HM6264-100 or Eqv.	1
U9	DP8391	1
U10	2VP5U9	1
U11	DP8390 Rev D	1
U14	PE64103	1
U15	DP8392 A-5	1
U16	PAL20L8 (DECODE1)	1
Y1	Crystal Oscillator Module 20 MHz +/−.01%	1
R1,R2,R3,R22	4.7K 5% carbon	4
R6,R7,R8,R9	39Ω 1%	4
R4,R5	510Ω 5% carbon	2
R10,R11,R12,R13	1.5K 5% carbon	4
R14	1.0k 1%	1
R17	1 MΩ ½W carbon	1
R15	Shorted	1
R18	RRX1-TDB Open	1
R16	RRX2-TBD Shorted	1
C1, C7–C17 (C10, C19 optional)	0.47 μF	13
C3, C21	47 μF ELECTROLYTIC	1
C5, C6	0.01 μF Ceramic	2
C4	0.01 μF Ceramic (600V)	1
JUMPERS	0.03 sp clips	16/board
J2	PANEL MT. BNC	1
J1	DB15	1
48-PIN SOLDERTAIL SOCKET	for U11	1
24-PIN SOLDERTAIL SOCKET	for U16	1
24-PIN AUGAT SOCKET	for U9	1

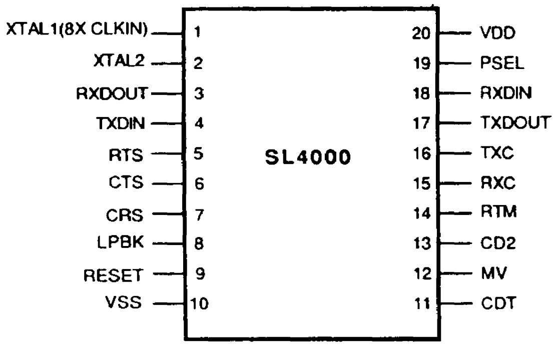
STARLAN INTERFACE SL4000

PRELIMINARY

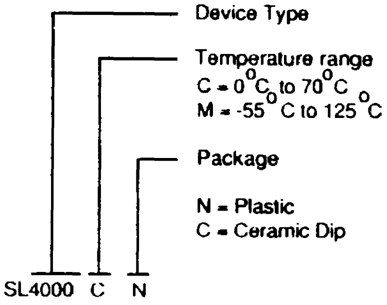
FEATURES

- o INTERFACE FOR THE LAN CONTROLLERS
- o UP TO 4 MBITS PER SECOND DATA RATE
- o MANCHESTER ENCODER AND DECODER
- o ON CHIP CRYSTAL DRIVER
- o DIGITAL PHASE LOCK LOOP FOR RECEIVE CLOCK RECOVERY
- o JITTER TOLERANCE OF 125 ns (At 1 Mb)
- o ON CHIP COLLISION DETECTION
- o LOCAL LOOPBACK FOR EASY DEBUGGING
- o OPTION TO REVERSE POLARITY OF SIX KEY SIGNALS
- o IMPLEMENTED IN HIGH-SPEED CMOS TECHNOLOGY

PIN OUT



ORDERING INFORMATION



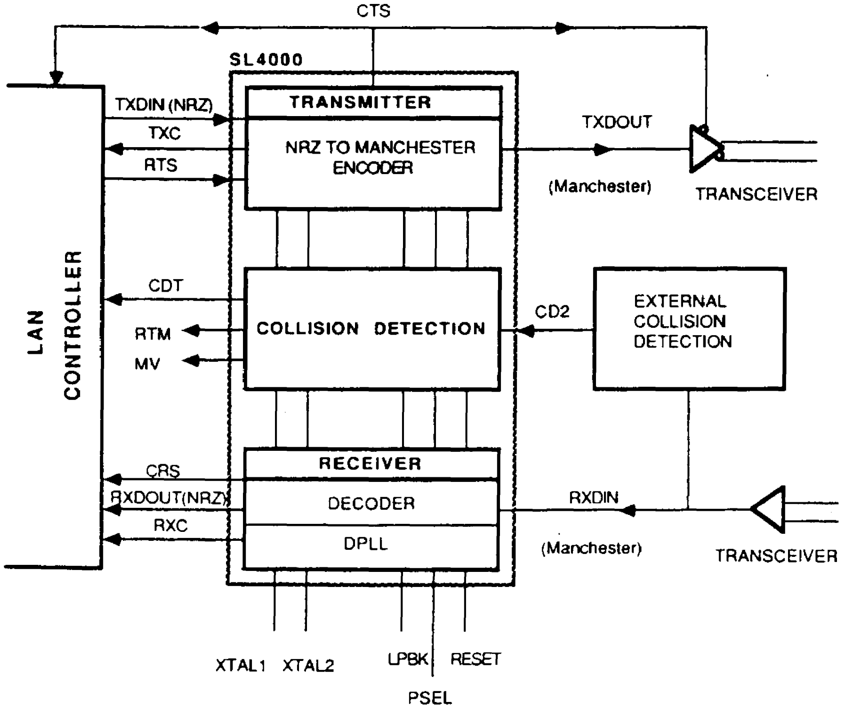


Figure1. Functional Block Diagram

May 1986

PIN DESCRIPTION

PIN	SYMBOL	TYPE	FUNCTION
1	XTAL1	INPUT	Crystal input (Or 8X clock - CMOS level)
2	XTAL2	OUTPUT	Crystal output
3	RXDOUT	OUTPUT	Receive data output (NRZ format)
4	TXDIN	INPUT	Transmit data input (NRZ format)
5	RTS	INPUT	Request to send
6	CTS	OUTPUT	Clear to send, Active LOW
7	CRS	OUTPUT	Carrier sense
8	LPBK	INPUT	When active, this pin loops back transmit data to received data
9	RESET	INPUT	Chip reset. Must be active for at least 1 TXC cycles for proper operation
10	VSS	GND	Ground
11	CDT	OUTPUT	Collision detect
12	MV	OUTPUT	Manchester violation, Active LOW
13	CD2	INPUT	Collision detect edge signal
14	RTM	OUTPUT	Retimed Manchester Data
15	RXC	OUTPUT	Receive clock
16	TXC	OUTPUT	Transmit clock
17	TXDOUT	OUTPUT	Transmit data output (Manchester format)
18	RXDIN	INPUT	Receive data input (Manchester format)
19	PSEL	INPUT	Selects polarity of pins 5, 7, 8, 11, 15 and 16
20	VDD	POWER	Power supply, 5V +/-10%

Note: All inputs are TTL compatible unless otherwise specified.

DESCRIPTION

The SL4000 is designed specifically for the 1Mbit per second CSMA/CD Local Area Network (LAN) commonly known as STARLAN. The device provides a glueless interface between industry standard LAN controllers (Like Intel's 82586, National's DP8390, Rockwell's R68802) and RS422 type transceivers.

The device consists of three major blocks (Figure 1):

i. TRANSMITTER: Performs NRZ to Manchester encoding including proper handshake.

ii. RECEIVER: Performs Manchester to NRZ decoding including clock recovery using 8X Digital Phase Lock Loop (DPLL).

iii. COLLISION DETECTOR: Dynamically checks the integrity of the data in both directions.

DEVICE OPERATION

RESET: Master reset input, when LOW, clears all internal counters and deactivates all output signals. When HIGH, the transmit and receive clocks are enabled.

TRANSMIT: The active RTS input from a controller signals begining of a transmit sequence. SL4000 waits for first TXDIN transition (No encoding is done until this time). This transition wakes up the encoder and it starts encoding NRZ data at TXDIN pin into the manchester format which is available at TXDOUTpin within 3 transmit clocks. The mid bit transition of TXDOUT is in synch with falling edge of TXC (When PSEL = HIGH). CTS output goes LOW 1/2 TXC cycles before first TXDOUT transition.

At the end of the transmit sequence the controller deactivates RTS within one clock of last transmit bit. SL4000 will deactivate CTS after 2 TXC cycles. The TXDOUT will be pulled HIGH (Idle state) after last encoded bit is transmitted. The delayed deactivating of CTS allows proper disabling of the transceiver.

RECEIVE: The receiver section is composed of two major blocks: 1) The 8X DPLL synchronizes the free running RXC to the incoming asynchronous data and 2) The DECODER converts manchester encoded data into NRZ format.

The RXDIN is held HIGH during idle and the DPLL is searching for any transition on this pin. When a transition occurs, the DPLL locks within 4 preamble bits (altemating 1's and 0's)to the incoming data. Upon lock, it checks the next two bits to be 01 or 10. A successful pattern check indicating detection of a legal carrier activates CRS. The manchester encoded data is now decoded and NRZ data is presented at RXDOUT. The maximum delay from the first preamble bit presented to SL4000 to the first decoded data bit out (and CRS activated) is no more than six bit times during normal operation.

The device continues to decode until it detects Legal End Of Frame- EOF (two consecutive missing transitions in the middle of the bit cell and RXDIN is HIGH). At this time SL4000 deactivates CRS and pull RXDOUT HIGH (idle) within 2.5 bit times, allowing for proper processing of the data in the pipeline. The device now waits for the next transition on RXDIN.

Retimed Manchester Signal is also available as an output and allows SL4000 to be used in HUB applications.

COLLISION DETECTOR:

i) MANCHESTER VIOLATION: During normal receive operation the SL4000 looks for missing mid-bit transitions (other than legal EOF) and declares a collision within 2.5 bits of a manchester violation.

ii) COLLISION AT START-UP: If a collision presence signal is presented to SL4000 without a preamble, the device, within 10 µs, will declare a collision and activate CDT as well as CRS indicating that a carrier is present but the data is not good. An end of frame will clear both CDTand CRS.

iii) ABSENSE OF ECHO: During normal operation in a STARLAN configuration TXDOUT should echo at RXDIN (and activate CRS) within 256 µS (One half slot time). If CRS is not activated within the 256 µs, a collision is declared by activating CDT. SL4000 provides enough margin between 256 µs and slot-time of 512 µs by activating CDT after 392 µs. If carrier is lost (after the 392 µs timeout) while RTS is still active, collision is declared and CDT activated. Removal of RTS clears CDT and the device resumes normal operation awaiting next RTS.

A LOW to HIGH transition on CD2 will activate CDT during transmission only (RTS active). Deactivating RTS also deactivates CDT. CD2 HIGH also disables the 392 µsec timer.

ADDITIONAL FEATURES

1. LOCAL LOOPBACK: When LPBK is active, the TXDOUT is internally routed to RXDIN, TXDOUT pin is held HIGH and the RXDIN pin is disabled. The 392 µsec timer also times out in 49 µsec.

2. POLARITY SELECT: When PSEL is HIGH, TXC and RXC (internally) sample related signals on the rising edge. Also CRS, CDT, RTS and LPBK are ACTIVE LOW. This allows easy interfacing with controllers that sample these signals on the falling edge. When PSEL is LOW, these six signals are complemented within SL4000.

OPTIONAL FEATURES

1. Transmit clock can be an input instead of an output. This allows SL4000 to be used with controllers that generate their own TXC.

2. FORCED COLLISION input CD1, when LOW, activates CDT. Only a HIGH on CD1 can deactivate CDT.

Note: These options are not available on standard SL4000 devices. Consult factory for details.

DELAYS and LOSSES

Description	Max bit delay	Max bit loss
1. First PREAMBLE TRANSITION to CRS active and RXDOUT valid.	6 (including loss)	4
2. First MANCHESTER VIOLATION to CDT active.	2.5	N/A
3. First COLLISION PATTERN (without PREAMBLE) at RXDIN to CDT and CRS active.	10 (including loss)	8
4. End Of Frame (EOF) to CRS inactive.	2.5	N/A
5. Last TXDOUT bit to CTS HIGH.	2.5	N/A
6. First (NRZ) PREAMBLE bit IN to First (Manchester) PREAMBLE bit OUT.	3	1

ELECTRICAL SPECIFICATIONS

D.C. SPECIFICATIONS:

TA = 0 - 70 C°, VDD = 5V +/-10%

Symbol	Parameter	Min	Max	Units	Remarks
VIL	Input LOW Voltage (TTL)	-0.5	0.8	V	
	(CMOS)	-0.5	1.5		
VIH	Input HIGH Voltage (TTL)	2.4	VDD	V	
	(CMOS)	3.5	VDD		
VOL	Output LOW Voltage (TTL)	-	0.4	V	IOL = 10 mA
	(CMOS)		0.1		IOL = 1µA
VOH	Output HIGH Voltage (TTL)	2.4	-	V	IOH = -10 mA
	(CMOS)	VDD-.05			IOH = -1 µA
IIN	Input Leakage Current	-10	10	µA	
IOZ	Output Leakage Current	-10	10	µA	
CIN	Input Capacitance		5	pF	
IDD1	Power Supply Current		5	mA	8XCLK=8MHz
IDD2	Quiescent Power Supply Current		100	µA	

AC Specifications:

TA = 0 - 70°C, VDD = 5V +/-10%

8X CLK Specifications

i) Crystal Frequency range (Pins X1, X2)
Max. 32 MHz Crystal, reasoning at fundamental frequency

Or

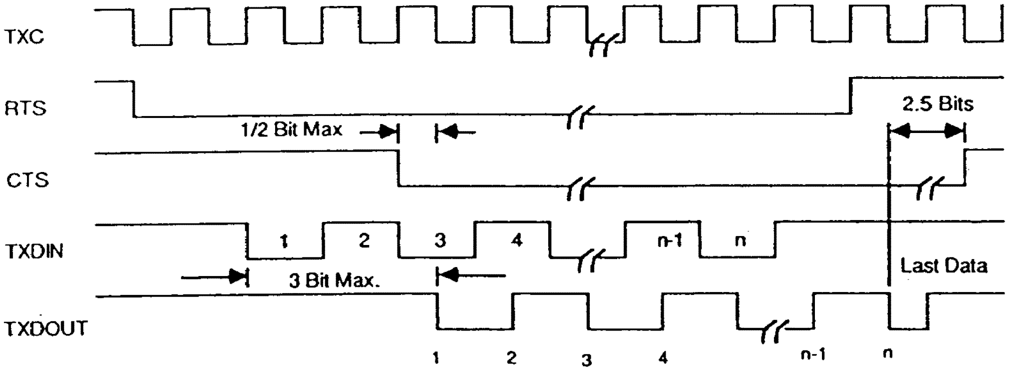
ii) CMOS Clock Input @ Pin X1
Max. 32 MHz

Symbol	Parameter	Min	Max	Units	Remarks
T1	8XCLK PERIOD	31.25		ns	125 ns for STARLAN
T2	8XCLK LOW Time	10		ns	
T3	8XCLK HIGH Time	10		ns	
T4	8XCLK Rise Time		5	ns	
T5	8XCLK Fall Time		5	ns	

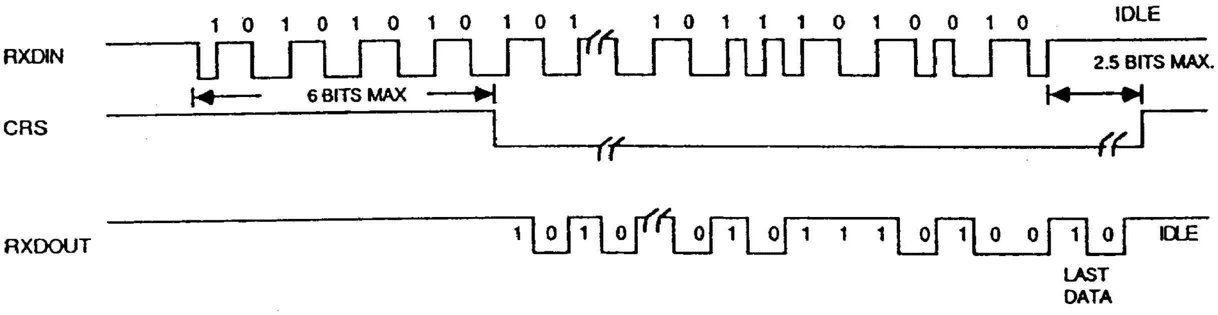
AC Specifications: TA = 0 - 70 °C, VDD = 5V +/-10%

Symbol	Parameter	Min	Max	Units	Remarks
T6	Rise Time Outputs (CMOS)		10	ns	CL = 15 pF
T7	Rise Time Outputs (TTL)		5	ns	CL = 15 pF
T8	Fall Time Outputs (CMOS)		10	ns	CL = 15 pF
T9	Fall Time Outputs (TTL)		5	ns	CL = 15 pF
T10	Rise Time at Inputs		10	ns	
T11	Fall Time at Inputs		10	ns	
T21	TXDOUT Transition to Transition	4	8	T1	
T22	TXDIN Hold to TXC	50		ns	
T23	TXDIN Setup to TXC	50		ns	
T24	TXDOUT Valid to TXC (Half bit/Full bit cell)	50		ns	
T25	TXC Period (Bit Time)	250		ns	(Bit Time = 1 µs at 1MHz for STARLAN)
T28	RTS to TXC Set-up	50		ns	
T29	RTS to TXC Hold	50		ns	
T30	CTS LOW to First TXDOUT Bit		0.5	T25	
T31	CTS HIGH from Last TXDOUT Bit		2.5	T25	
T32	RXDIN Half Bit Cell	4		T1	
T33	RXDOUT to RXC Hold	250		ns	
T34	RXDOUT to RXC Setup	250		ns	
T35	RXDIN to CRS		6	T25	
T36	CRS Delay from Last Data Bit		100	ns	
T37	Receive Data Jitter Tolerance From Center of Bit Cell		+/- 1	T1	(125 ns for 1 MBit STARLAN)
T38	Transmit Data Jitter		1	ns	

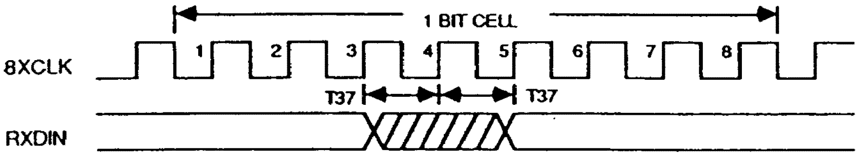
AC Timing Diagrams



TRANSMIT SEQUENCE



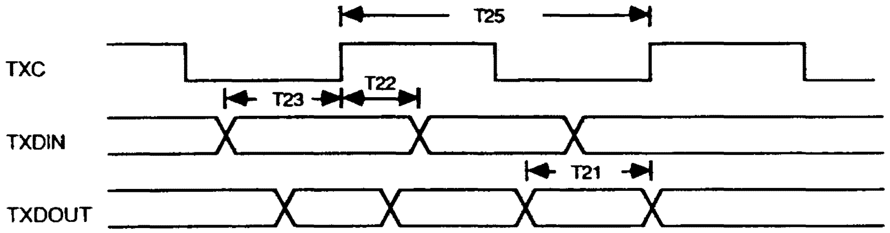
RECEIVE SEQUENCE



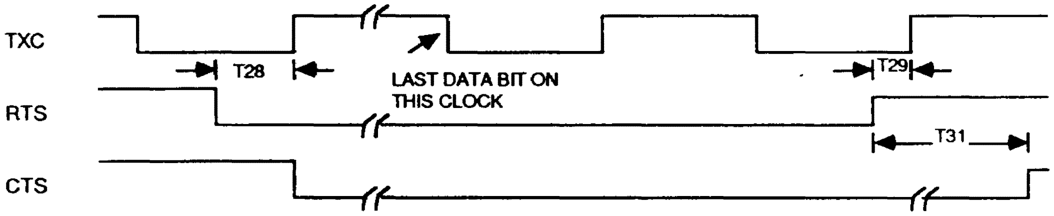
RECEIVE DATA JITTER TOLERANCE

(+/- 125 ns at 1MB Data rate)

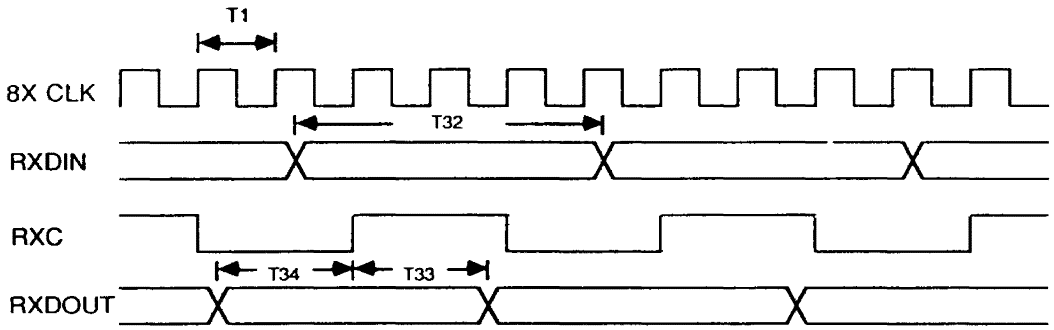
NOTE: All timings assume PSEL = HIGH.



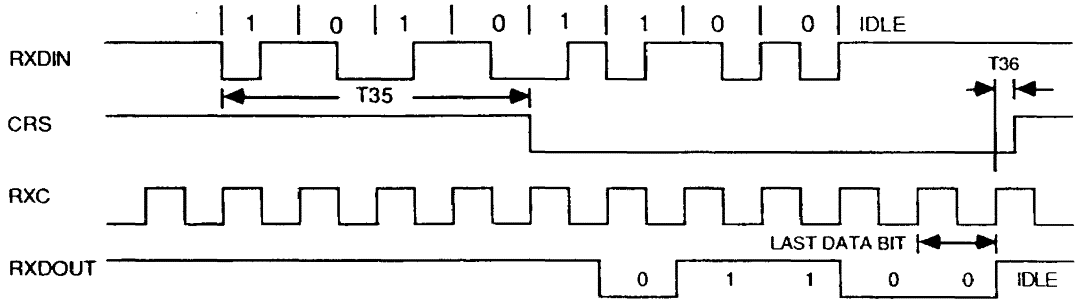
TXC, TXD TIMING



RTS, CTS TIMING



RXD, RXC TIMING



RXD, RXC CRS TIMING

SLI reserve the right to make changes in specifications at any time without notice. The information furnished by SLI in this publication is believed to be accurate and reliable. However, no responsibility is assumed by SLI for its use; nor for any infringements of patents or other rights of third parties resulting from its use.

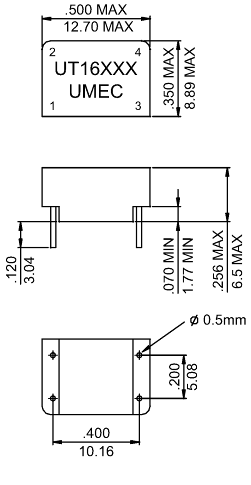
ISOLATION TRANSFORMERS AND DATA BUS INTERFACE MODULE FOR STARLAN APPLICATIONS

¡· IEEE 802.3 (1 base 5) compatiable.
¡· Single and dual low profile packages.
¡· Surface mount package option.
¡· High isolation voltage to comply with international safety requirement for LAN's. (ECMA 97)
¡· Designed for hub and node use.
¡· Fast rise time available.
¡· 4 wire filters available.
¡· Modules provide galvanic isolation and common mode suppression for the data bus interface.

ELECTRICAL SPECIFICATIONS @25^{0}C

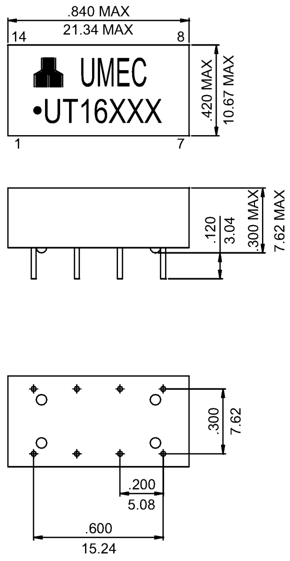
Part No.	Turns Ratio (±5%)	Sine Wave Inductance (mH±20%)	Pulse[4] Inductance (mH Min)	Primary ET-Constant (V-µs Min)	Rise Time (ns Max)	Pri/Sec Cw/w (pf Max)	Pri-Sec Leakage Inductance (µH Max)	Primary DCR (ohms Max)	Hi-pot Test (Vrms)[5]	Package Style
UT16352	1:1	2.5	2.0	20	100	25	8	1.0	2000	A
UT16552	1:1	2.5	2.0	20	25	40	1.1	1.0	2000	A
UT16382[8]	1:1	2.5	2.0	20	100	25	8	1.0	2000	B
UT16582[8]	1:1	2.5	2.0	20	25	40	1.1	1.0	2000	B
UT16514	1:1:1:1	0.03[7]	N/A	N/A	N/A	30	0.3	0.13	2000	C
UT16573	1:1	2.5	2.0	20	100	25	8	1.0	2000	D
UT16571	1:1	2.5	2.0	20	25	40	1.1	1.0	2000	D

PACKAGE A

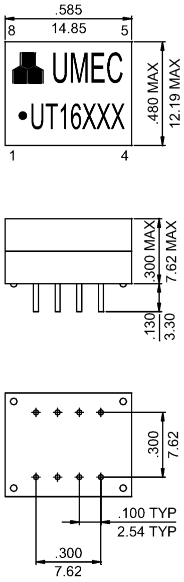


.XXX=¡ .010 inches
.XX=¡ .25 mm

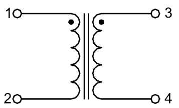
PACKAGE B



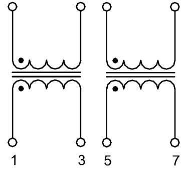
PACKAGE C



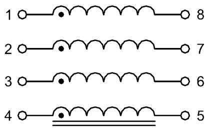
SCHEMATIC A



SCHEMATIC B



SCHEMATIC C



*Specifications are subject to change without prior notice.