LCG NI Port Architecture Specification

COMPANY CONFIDENTIAL

Mike Wolinski
Thomas G. Arnold
MR01-2 E/18
DTN 231-5707

10-NOV-83
Revision 6.0

1.0 GOALS

This document describes the software interface to control an LCG NI port. The LCG NI port will implement the necessary functionality to meet the required DEC corporate specs for devices connected to the NI. These include, but are not limited to, the bluebook Ethernet specification, the corporate NI Product Architecture, the DNA NI Data Link Specification, and the Low Level Maintenance Operation (MOP) specification. Port services and functionality are described.

2.0 NON-GOALS

It is not a goal of this specification to describe the physical channel, or to resolve higher level network issues. No specific network architecture beyond that defined in the data link Ethernet specification is implied.

3.0 NOTATIONAL CONVENTIONS

All numbers in this specification will be in decimal unless explicitly stated otherwise. Numbering of bits in words conforms to the DEC-20 standard of numbering the left most (highest order) bit as 0, and proceeding to the right from there. The number of the right most (least significant) bit in a PDP-20 word is 35.

4.0 REFERENCES

The reader is assumed to be familiar with the the following documents:

1. LCG Network Interconnect Adaptor (NIA) Interface Specification. Author Bernie Hall and John Halstead.

2. NI Node Product Architecture. Version 1.0 Author: P.J. Nesbeda. This document describes the minimum requirements of a DEC NI node.

3. DNA Low Level Maintenance Operations Architecture. Version 3.0.0 Authors: Bob Stewart and Ken Chapman This documents describes the maintenance features available in most NI nodes.

4. DNA NI Data Link Architecture. Version 1.0 Author: Bob Stewart. This document integrates the "Ethernet" spec into DNA.

5. VAX-11 NI Port Architecture. Author: William Strecker, Bruce Mann. This documents defines the architecture of the NI port for the VAX family of processors. This document is the model for the LCG NI port architecture.

6. The Ethernet - A Local Area Network - Data Link Layer And Physical Layer Specifications. Version 2.0 Author: DEC, Xerox, Intel. This is THE description of the NI itself.

5.0 INTRODUCTION

The NI is a multiaccess serial interconnect that allows up to 1024 different stations to share a common 10 Megabit bus and communicate by exchanging data packets. The maximum length of the NI interconnection is 2.5 kilometers (1.5 miles). The transmission medium is a coaxial cable, using base band signaling. The arbitration protocol used by the NI is carrier sense multiple access with collision detect (CSMA/CD.) The service provided by the NI is a datagram-class of service in which the data packets are delivered on a best-effort basis; no confirmation of delivery is made, and no guarantee of sequentiality or non-duplication is made. Higher levels of network software must provide these services based upon the basic Ethernet service offered by this port. A full description of the NI itself appears in the Ethernet specification referred to above.

5.1 Port Usage

The NI20 will interface between the NI and the KL10 E/Cbus.This port provides the Data Link and Physical Channel layers of the ISO (International Standards Organization) OSI (Open Systems Interconnection) reference model.

5.2 Port Services

The NI port performs queued retrieval of commands, queued reporting of received packets, packet transfer to and from host memory, packetization, framing, CRC generation and checking, Ethernet arbitration, transmission retries, and error reporting. The NI also does filtering for multi-cast addresses, and broadcast addresses, as well as physical destination addresses. Protocol type filtering is done. Free queue entries for enabled protocol types are obtained from a list of protocol types and associated free queue pointers.

Received packet address filtering is done in the following manner: In order to receive a datagram, the destination field of the datagram must pass the received address filter. If the destination address is -1 (all ones), then the message is a BROADCAST message, and the port accepts it. If the destination address has bit 47 a zero, then it is a physical address; If this destination address matches the physical address of the link, the port accepts it. If the destination address has bit 47 a one, then it is a multi-cast address. If this destination address matches one of the multi-cast addresses enabled for the port, the datagram is accepted.

If the packet is accepted by the port address filter, Protocol Type filtering occurs as follows: When a message is received, the protocol type is checked against the table of enabled protocol types. If a match is found in this table, the associated free list is used to obtain the queue entry to store the received packet. This feature allows effective queue management by the port driver. If a match is not found, the required queue entry is obtained from the Unknown Protocol Type free list queue anchored in the PCB. If the queue checked in order to get the free entry is empty, the datagram is discarded.

5.3 NI Packet Format

The basic format of a frame (packet) of data which appears on the NI wire follows:

```
+-------------------------+
|                         |
|   DESTINATION ADDRESS   | 6 BYTES
|                         |
+-------------------------+
|                         |
|     SOURCE ADDRESS      | 6 BYTES
|                         |
+-------------------------+
|                         |
|     PROTOCOL TYPE       | 2 BYTES
|                         |
+-------------------------+
|                         |
|          DATA           | 46-1500 BYTES
+-------------------------+
|                         |
|  FRAME CHECK SEQUENCE   | 4 BYTES
|                         |
+-------------------------+
```

The following definitions apply:

1. Destination address - These 6 bytes contain the address of the port that is the intended recipient of this frame.

2. Source address - These 6 bytes contain the address of the port that transmitted this frame.

3. Protocol type - These 2 bytes defined the format and content of the remainder of the packet. MCP protocol messages undergo special processing, described later

4. Data - These bytes are the actual data bytes intended for the receiver.

5. Frame check sequence - This is the 4 byte CRC field that is used to guarantee the uncorrupted reception of the packet.

6.0 ARCHITECTURE OVERVIEW

The LCG NI port model is a single port communicating with a single port driver which provides the multiplexing and demultiplexing services necessary for many users to share the NI wire. The port and port-driver software communicate with each other through the use of queues and a Port Control Block.

6.1 Port Addresses

All host memory addresses used by this port are physical addresses in KL10 memory. All pointers in the Port Control Block, the Protocol Type Table, the Multi-cast Address Table, the queue entries, and the free standing queue headers refer to physical addresses. Reserved for software words are provided for the use of the driver program. One use may be to hold the virtual address counterpart for these structures.

6.2 Port Control Block

The Port Control Block is used to anchor the queues at a known point in the host memory and to provide certain initial parameters to the port. The queues are used to pass commands from the port-driver software to the port for either local execution or for transmission over the NI wire. The queues are also used by the port to pass responses back to the port driver software and to deposit packets received over the NI wire.

The Port Control Block (PCB) is a data structure based in the host memory that allows the sharing of the queues by the port driver and the port. The Port Control Block is pointed to by port register 2, the PCB Base register.

The port is informed of the location of the PCB at micro code intialization time by the port driver software. When this is detected by the port, it will cache the following variables from the PCB: the unknown protocol type queue entry length, the portocol type table starting address, and the multi-cast address table starting address.

Both the host port driver and port read and write locations in the PCB. There is exactly one PCB for each NI port controlled by the host system. The PCB is the main control structure for the NI port. It anchors all of the tables and queue structures. The PCB contains queue headers to anchor the command queue, the response queue, and the unknown protocol type free queue. Base pointers to the Multi-cast address table, and the protocol type table are located in the PCB. In addition, an error logout area, and several free words provided for the use of the driver program are included in the PCB. The reseved words will never be altered or examined by the NI port. The error logout area may be written at any time by the port to record an error event.

6.2.1 Queue Headers - A queue consists of a queue header which anchors the queue and a number of entries, each occupying a spot on the queue. All LCG NI queues are doubly linked structures. The queue header and each queue entry contain a forward link (FLINK) and a backwards link (BLINK.) The forward link of a queue header points at the first entry of the queue. The forward link of a queue entry points to the next entry of the queue, if any. The backwards link of a queue header points at the last entry of the queue. The backwards link of a queue entry points back at the entry before the entry on the queue, if any. If a flink does not point to a queue entry, it points at the queue header flink. If a blink does not point to a queue entry, it points at the queue header flink.

Queue headers anchor a queue structure. A queue header may be located in the PCB, or located in host memory as a free standing structure. A queue header is composed of a reserved word, a FLINK, a BLINK, and a Queue Length. The FLINK (forward link) points to the first word, the FLINK, of the first entry of the queue. The BLINK points to the FLINK word of the last entry of the queue. The first and last entries may be the same entry. If there are no entries on the queue, the queue header FLINK points to itself.

The Queue Length word, where applicable, specifies the length in words, of the entire entry, including the FLINK, BLINK, Command header, and data words. The queue length applies to every entry on the queue. All queue entries on one queue must be the same length.

6.2.2 Queue Interlocks -

The NI20 requires special KL10 microcode support to allow the NI20 to perform a memory increment operation using read-pause-write memory references. This is needed to allow the port to interlock the queues.

There is a separate interlock word for each queue. When a queue is available, the corresponding interlock word has a value of -1. When either the operating system or the port want to interlock the queue, they must perform a non-interruptable increment-store-test operation, such as an AOSE. If the incremented location has a value of zero, then the queue has been successfully interlocked and the process may now manipulate the queues. If the incremented value is greater than zero, then the queue is not available. The interlock word should not be set back to zero. When the process is finished with the queues, the interlock word must be set back to -1 (all ones). This marks the queue as available. Both the port driver and the port microcode are responsible for leaving the queues in a well defined state.

The PCB must be allocated starting on a four word boundary. The format of the Port Control Block is illustrated below:

DEC	PCB Format	OCT
0	Command Queue Interlock	0
1	Command Queue FLINK	1
2	Command Queue BLINK	2
3	Reserved for Software	3
4	Response Queue Interlock	4
5	Response Queue FLINK	5
6	Response Queue BLINK	6
7	Reserved for software	7
8	Unknown Protocol Type Free Queue Interlock	10
9	Unknown Protocol Type Free Queue FLINK	11
10	Unknown Protocol Type Free Queue BLINK	12
11	Unknown Protocol Queue Entry Length	13
12	Reserved for Software	14
13	Protocol Type Table starting address	15
14	Multi-cast Address Table starting address	16
15	Reserved for Software	17
16	Error Logout 0	20
17	Error Logout 1	21
18	RSVD	22
19	RSVD	23

```
          +-------------------------------------------------------------+
   20     |                     PCB Base Address                        | 24
          +-------------------------------------------------------------+
   21     |                    Reserved to Port                         | 25
          +-------------------------------------------------------------+
   22     |                    Reserved to Port                         | 26
          +-------------------------------------------------------------+
   23     |                  Channel Command Word                       | 27
          +-------------------------------------------------------------+
   24     |                    Reserved to Port                         | 30
          +-------------------------------------------------------------+
```

The Error Words (words 20,21) are written by the port when it encounters fatal errors associated with Queue manipulation. This error reporting strategy requires the port to write as much information as possible directly into the host memory. This approach requires the smallest subset of port hardware and microcode to be working to report these errors.

The information in these words provides sufficient data for the port driver to determine the type of error and where the error occurred. When the error is detected, the port will write the contents of the Error Words in the PCB, enter the Disabled State, and generate a host interrupt.

The format of Error Word 0 is:

```
    0       1       2       3    4                        11 12              35
+-------+-------+-------+--------------------------+-----------------+
|  CMD  |   Q   | RESP  |        MUST BE ZERO      | FLINK ADDRESS   |
+-------+-------+-------+--------------------------+-----------------+
```

BITS	NAME	DESCRIPTION
0	CMD	Error occurred while reading a command queue entry. The queue with the error is in the Q bits, bits 1-2.
1-2	QUEUE	This is the command queue that had the error. These bits are only valid if the CMD bit is on.
3	RESPONSE	This bit is on if the error occurred while the port was attempting to build a response queue entry.

4-11	MBZ	These bits will be zero.
12-35	FLINK ADR	This is the address of the FLINK word of the queue entry in question.

Error word 1 contains the API function word that the port processor used to access memory when the memory error occurred. This word is written here in the same format as it should have appeared on the EBUS. The format of this word is:

```
 0    2 3    5  6  7 12 13                            35
+------+------+---+----+-------------------------------+
| Addr | Func | Q | 0  |       Physical Address        |
| Code |      |   |    |                               |
+------+------+---+----+-------------------------------+
```

Word 24 of the PCB is the address of the first word of the PCB; the NI20 has no other way of finding the PCB.

Word 27 is reserved for the Channel Command Word. The port will write a CCW-style word here when it wishes to transfer data over the KL10 CBus. The port driver is responsible for writing a Channel Jump Word into the appropriate EPT location corresponding to the KH20 backplane slot that the NI20 is installed in.

Word 25 is always reserved to the port microcode for its use; the port driver should never write this location nor depend upon its value.

When the NI20 is being initialized, the port driver must set up the channel to transfer the contents of the PCB into the port. This is done by setting up a CCW to transfer 3 words starting with word 20 of the PCB from KL10 memory to the channel. The port will start the channel and will read the contents of these locations. This provides the port with the base of the PCB, and its PI assignment.

It is important to realize that since the port will be using the channel to transfer large blocks of data, the channel will be writing logout information into the EPT. An error that the channel discovers will be reported in the usual manner via the EPT.

6.2.3 Command Queue -

The command queue is the basic communication structure for the LCG NI. Through it, all commands are passed to the port interface. A command consists of a queue entry, with a FLINK (Forward LINK), pointing to the next command in the queue, if any, and a BLINK, (Backward LINK) pointing to the previous command in the queue, if any. From the OPCODE field of the command queue entry is obtained what function is to be performed. The queue entry contains the information needed to process the command. Commands are delinked from the beginning of the queue by the port, and executed. Commands are executed in the order in which they are placed in the command queue. The driver places command entries to be executed on the tail of the command queue. If the queue was previously empty when this occurs, the driver writes the Command Queue Available (CQA) bit in the CSR register. The length of the command entry is inferred from its format, and the function being performed.

6.2.4 Unknown Protocol Type Free Queue -

This queue is a linked list of free queue entries to be used to report to the driver all received packets which passed address filtering, and which are addressed to a protocol type which is not enabled in the Protocol Type Table. The driver links new free entries onto the end of this queue as it obtains them. The port delinks these entries when it must build a response, and cannot use one of the enabled protocol type free queues.

The unknown protocol queue entry length word specifies, in words, how long the entries on the queue are. This length included the FLINK, BLINK, Reserved and Opcode words of the entry. These different queues are provided to help insure that heavy traffic on one protocol type will not disturb the availability of ready queue entries for other protocol types. In addition, by having different queues for different sorts of messages, all queue entries need not be the maximum length for Ethernet messages; protocol types which use only a limited size packet can allocate more entries of smaller size, and yet run with other protocol types which require larger messages.

It is recommended that queue entries for the Unknown Protocol Type Free Queue be large enough to hold the largest legal Ethernet packet. This is 388 (decimal) words, assuming industry compatible is the packing mode. This figure includes the flink, blink, etc. of the send datagram non-BSD command format.

6.2.5 Response Queue -

The Response Queue is used by the port to submit to the driver packets which were accepted by the packet address filter over the NI wire, or which resulted from a completed command, or an error packet. The port links such packets onto the end of the queue. If the queue was previously empty when this occurs, an interrupt is submitted to the host, to inform it of the presence of a new response. This queue is accessed according to the interlocked queue protocol for the KL10.

6.2.6 Error Logout Area -

6.2.7 Protocol Type Table Starting Address -

This word points to the beginning of the protocol type table. This table, described elsewhere, lists the protocol types to be accepted, and pointers to appropriate free queue headers, from which free entries to hold received responses for that protocol type. The PTT must be allocated beginning on a four word boundary. The protocol types listed in this table do not come into effect until after the execution of a Load PTT command. Once this table has been allocated and its address specified in the PCB the location of the table cannot be changed.

6.2.8 Multi-Cast Address Table Starting Address -

This word points to the beginning of the multi-cast address table. This table lists the multi-cast addresses to which the port is to respond. Multi cast addresses on the NI wire that are not present in this table will be ignored. The addresses in this table come into effect only after the execution of a Load MCAT command. This table must be allocated beginning on a four word boundary. Once this table has been allocated and its address specified in the PCB, its location cannot be changed.

6.3 Queue Structures

Queued structures comprise the basic communication between the host driver and the NI port. This section describes these memory structures.

6.3.1 Queue Headers -

The format ofthe queue header is illustrated below:

```
+-----------------------------------------------------+
|               Queue Interlock Word                  |
+-----------------------------------------------------+
|                   Queue FLINK                       |
+-----------------------------------------------------+
|                   Queue BLINK                       |
+-----------------------------------------------------+
|               Queue Entry Length                    |
+-----------------------------------------------------+
```

Queue Entry Length

The Queue Entry Length specifies the length, in words, of the queue entries on the queue. This queue length included the FLINK, BLINK, reserved, and opcode words. All entries on a queue are of the same length.

Reserved For Software

The reserved word may be used by the driver program for whatever purpose it wishes. The independent queue header must be allocated on a four word boundary. Independent queue headers are subject to the same rules for interlocking that the PCB queues enjoy.

6.3.2 Queue Entries -

Queue entries are the structures out of which commands and responses are built. They possess forward and backward links, to enable the construction of a doubly-linked list. The format of a queue entry is given below:

```
+-----------------------------------------------------------+
|                          FLINK                            |
+-----------------------------------------------------------+
|                          BLINK                            |
+-----------------------------------------------------------+
|                  Reserved for Software                    |
+-----------------------------------------------------------+
|               Operation Code (Entry type)                 |
+-----------------------------------------------------------+
|                      Queue Data 0                         |
+-----------------------------------------------------------+
|                      Queue Data 1                         |
+-----------------------------------------------------------+
                             .
                             .
                             .
+-----------------------------------------------------------+
|                      Queue Data N                         |
+-----------------------------------------------------------+
```

The FLINK points to the next entry of the queue (Forward LINK). BLINK, on the other hand, points to the previous queue entry. If there is no next entry, then the FLINK points back to the queue header FLINK. If there is no previous entry, the BLINK points back to the FLINK word of the queue header.

The word reserved for software will never be altered or examined by the LCG NI port. The Operation Code specifies the type of the entry; i.e., Send Datagram, Load MCAT, etc. Words following the Operation code are type dependent.

6.3.3 Protocol Type Table -

The Protocol Type Table specifies a set of protocol types, with their associated free queue pointers that are considered enabled by the port. When a message which passes the received address filter is detected, this table will be examined. Thus, for a multi-cast packet to be received into a protocol type queue (either known or unknown) its destination address must be enabled in the multi-cast address table. If the protocol type of a received packet matches an entry in the PTT, then the associated free queue of that PROTOCOL TYPE entry is used to supply the free queue entry used to store the incoming packet. A protocol type is not considered unless the corresponding enable bit (sign bit) is on. It is required to have the protocols arranged in the table in ascending numerical order. This allows the microcode to stop searching the table after the first time that the protocol type of the incoming packet is less than an examined entry in the protocol type table.

The number of protocol types allowed for a particular version of the NI microcode is specified in the NI read station information. The port driver must read this number in order to discover how many entries are available for both MCAT entries, and PTT entries. The Protocol Type Table must be allocated beginning on a four word boundary. The table must be built with the number of entries specified in the NI read station information, even if some of the entries are not enabled. Listed below is the general format of the Protocol Type Table:

```
                          Protocol Type Table
                                                                  OCT
   ---+----------------------------------------------------------+---
      |   0    |<1---------------15>|        <16-31>         |  |  0
      | Enable |        MBZ         | Protocol Type value 0 |  |
      +----------------------------------------------------------+
      |               FREEQ 0 queue header address              |  1
      +----------------------------------------------------------+
      |                 Reserved for Software                   |  2
   ---+----------------------------------------------------------+---
      |   0    |<1---------------15>|        <16-31>         |  |  3
      | Enable |        MBZ         | Protocol Type value 1 |  |
      +----------------------------------------------------------+
      |               FREEQ 1 queue header address              |  4
      +----------------------------------------------------------+
      |                 Reserved for Software                   |  5
   ---+----------------------------------------------------------+---
      |   0    |<1---------------15>|        <16-31>         |  |  6
      | Enable |        MBZ         | Protocol Type value 2 |  |
      +----------------------------------------------------------+
      |               FREEQ 2 queue header address              |  7
      +----------------------------------------------------------+
      |                 Reserved for Software                   |  10
   ---+----------------------------------------------------------+---
      |   0    |<1---------------15>|        <16-31>         |  |  11
      | Enable |        MBZ         | Protocol Type value 3 |  |
      +----------------------------------------------------------+
      |               FREEQ 3 queue header address              |  12
      +----------------------------------------------------------+
      |                 Reserved for Software                   |  13
   ---+----------------------------------------------------------+---
      |   0    |<1---------------15>|        <16-31>         |  |  14
      | Enable |        MBZ         | Protocol Type value 4 |  |
      +----------------------------------------------------------+
      |               FREEQ 4 queue header address              |  15
      +----------------------------------------------------------+
      |                 Reserved for Software                   |  16
   ---+----------------------------------------------------------+---
      |   0    |<1---------------15>|        <16-31>         |  |  17
      | Enable |        MBZ         | Protocol Type value 5 |  |
      +----------------------------------------------------------+
      |               FREEQ 5 queue header address              |  20
      +----------------------------------------------------------+
      |                 Reserved for Software                   |  21
   ---+----------------------------------------------------------+---
```

```
                              .                                   .
                              .                                   .
                              .                                   .

   ---+-----------------------------------------------------------+---
      |   0    |<1---------------15>|       <16-31>          |   |  M-2
      | Enable |        MBZ         | Protocol Type value N  |   |
      +-----------------------------------------------------------+
      |               FREEQ N queue header address                |  M-1
      +-----------------------------------------------------------+
      |                  Reserved for Software                    |  M
   ---+-----------------------------------------------------------+---
```

The LCG NI caches the PTT internally when the LOAD PTT command is issued. The driver is free to alter the PTT whenever the LOAD PTT command is not pending. The protocol values within the table must be assembled in ascending numerical order. Only the binary value of the protocol type itself is considered in the sorting order. Non enabled protocol types are ignored.

If a protocol type is received which is not in this table, the message will be placed onto the Unknown Protocol Type queue, anchored in the PCB.

If the port is enabled without having loaded the PTT after initialization, all protocol types are disabled, except for those MOP protocol types which are implied active by the setting of the RMOP mode bit.

Enable

This bit, when set, indicates that the associated protocol type is valid, and should be considered when the PTT is cached.

Protocol Type Value

This variable specifies the value of protocol type that should be placed on the associated queue when a message of the protocol type is received.

Note that the free queue indirection scheme such as the one discussed above allows the sharing of free queue lists between protocol types.

Queue Header Address

This word contains the physical address of the FLINK word of the associated free standing queue header block.

6.3.4 Buffer Descriptors -

At times it is desirable to have different portions of a datagram be in different packing modes, or to allow different portions of a datagram to be built at different times without incurring substantial overhead in copying the datagram from buffer to buffer. An example of this is when lower layers of network software wish to prepend data to a user's data in order to facilitate flow control, routing, session control, etc. This feature is accomplished through the mechanism of buffers.

The use of buffers for a commands is activated by setting the BSD bit of the flags byte.

A buffer consists of a list of segments of memory, each segment being physically contiguous, and resident within physical memory. Different segments of a buffer are not assumed contiguous, and may lie anywhere within the physical address space of the host. It is assumed in normal use that different segments of a buffer are unique (i.e., they do not overlap) within a host, and that each segment is in only one packing mode. A buffer is described by a list of Buffer Segment Descriptors (BSDs).

Each buffer segment descriptor describes a single contiguous piece of a buffer. Each descriptor is a four word block, allocated on a four word boundary, and built by the driver in physical host memory. In each of these blocks is a pointer to the next descriptor block (if any), a pointer to the segment of the buffer so described, and a field indicating what packing mode the segment is in. In addition, within each block is a field giving the size of the buffer segment in bytes.

A buffer is referenced by giving the physical address of the first word of the first BSD in the chain of BSDs. This address is called out in the SEND DATAGRAM command packet, when the flags byte of that command indicates that buffer descriptors are in use. The format of the SEND DATAGRAM command is effectd by the setting of this flags bit.

Note that this is unlike the CI20, wherein buffers are called out by name and a key, and are referenced by indexing within a table of buffer header blocks in order to obtain the address of the chain of buffer segment descriptors. The BSD format is specified below:

```
                      (buffer segment descriptor)
 +-----------------------------------------------------------+
 |         |      <6>      |       |           <12-35>       |
 |   MBZ   | Packing Mode  |  MBZ  |  Segment Base address   |
 +-----------------------------------------------------------+
 |                                 |        <12-35>          |
 |            MBZ                  |    Next BSD address     |>---+
 +-----------------------------------------------------------+    |
 |                                          |<20----------35>|    |
 |                    MBZ                   | Segment Length |    |
 +-----------------------------------------------------------+    |
 |                Reserved for use by software               |    |
 +-----------------------------------------------------------+    |
                                                                  |
                                                                  |
                                                                  |
                      (buffer segment descriptor)                 |
 +-----------------------------------------------------------+    |
 |         |      <6>      |       |           <12-35>       |    |
 |   MBZ   | Packing Mode  |  MBZ  |  Segment Base address   |<---+
 +-----------------------------------------------------------+
 |                                 |        <12-35>          |
 |            MBZ                  |    Next BSD address     |
 +-----------------------------------------------------------+
 |                                          |<20----------35>|
 |                    MBZ                   | Segment Length |
 +-----------------------------------------------------------+
 |                Reserved for use by software               |
 +-----------------------------------------------------------+
```

Packing Mode 0<6>

This field describes the packing mode that the buffer segment pointed to is in. The packing moces, with the associated bit values are listed below:

1. 0 - Packing mode = Industry Compatible

2. 1 - Packing mode = High Density

For an extended discussion of the packing modes, please consult the section on packing modes.

Segment base Address 0<12-35>

This field gives the physical address of the beginning of the buffer segment. The buffer segment must begin on a whole word boundary.

Next BSD address 1<12-35>

The next BSD address points to the first word of the next buffer segment descriptor in the chain. If the next BSD field is zero, then there is no next segment descriptor.

Segment Length 2<20-35>

This field gives the length in bytes of the buffer segment pointed to. Note that the total legal length of the text portion of an Ethernet packet is 1500 bytes.

It is allowed that BSDs may be built in the queue entries on the command queues. Indeed, it is assumed that this is the standard use. BSDs must be allocated on a four word boundary.

6.4 Multi-cast Address Table

The MCAT is the Multi-Cast address table. This table specifies which multi-cast NI addresses are to be accepted by the LCG NI address filter. Note that an address is not considered for comparison unless the associated enable bit is on.

The MCAT is cached internally when the Load MCAT command is issued. The port driver should refrain from altering the MCAT when such a load command is pending, that is, after the command is linked onto the command queue and before the command is linked either onto the response queue or the unknown protocol type free queue. The port will not examine the MCAT until such a command is issued. The port will never change data in the MCAT, although it will read all entries whenever the load command is issued.

In ALL MULTI-CAST mode, all received packets with the MULTI-CAST bit set in the destination address will pass received packet address filtering. In order to enable ALL MULTI-CAST mode, the driver must set the AMC bit in the write NI Station Address command for the KL10. The following is the arrangement of a pair of words that specify one multi-cast address. Byte 0 is the first byte received or transmitted over the NI wire. Byte 5 is the last byte received or transmitted. Therefore, bit 31 of the first word of the pair is the first bit on the wire, and is thus the MULTICAST bit.

Binary Order Of Multi-cast Address

```
0.........7 8..........15 16........23 24........31  32-35
+--------------------------------------------------------+
| Byte 3    |    Byte 2   |  Byte 1    |  Byte 0   | MBZ |  Word 0
+--------------------------------------------------------+
                                                ^
                                         multicast bit

   0   1..............15 16........23 24........31  32-35
+--------------------------------------------------------+
| ENA |    MBZ          |    Byte 5   | Byte 4   | MBZ |  Word 1
+--------------------------------------------------------+
```

The following illustrates the format of the MCAT. Entries that have the enable bit off when the MCA is cached are not used in address comparison. Each MCAT entry conforms to the binary order of NI addresses format shown above.

MCAT Format

```
<  1 ><2 . . . . . . . . . . . . . . . . . . . . 31> <32:35>
+-------------------------------------------------------+
|              Multi-cast address 0, Word 0     | MBZ  |
+-------------------------------------------------------+
| ENA |        Multi-cast address 0, Word 1     | MBZ  |
+-------------------------------------------------------+
|              Multi-cast address 1, Word 0     | MBZ  |
+-------------------------------------------------------+
| ENA |        Multi-cast address 1, Word 1     | MBZ  |
+-------------------------------------------------------+
|              Multi-cast address 2, Word 0     | MBZ  |
+-------------------------------------------------------+
| ENA |        Multi-cast address 2, Word 1     | MBZ  |
+-------------------------------------------------------+
|              Multi-cast address 3, Word 0     | MBZ  |
+-------------------------------------------------------+
| ENA |        Multi-cast address 3, Word 1     | MBZ  |
+-------------------------------------------------------+
|              Multi-cast address 4, Word 0     | MPZ  |
+-------------------------------------------------------+
| ENA |        Multi-cast address 4, Word 1     | MBZ  |
+-------------------------------------------------------+
                                  .
                                  .
                                  .
    +---------------------------------------------------------+
    |             Multi-cast address N-1, Word 0       | MBZ  |
    +---------------------------------------------------------+
    | ENA |       Multi-cast address N-1, Word 1       | MBZ  |
    +---------------------------------------------------------+
    |             Multi-cast address N, Word 0         | MBZ  |
    +---------------------------------------------------------+
    | ENA |       Multi-cast address N, Word 1         | MBZ  |
    +---------------------------------------------------------+
```

The table is always considered full (i.e., the port will always read the entire table), although word pairs without the enable bit on are not considered. The MCAT must be allocated beginning on a four word block. The number of MCAT entries allowed is specified by the NI Configuration port register.

Note that the high order bit of word 1 of each MCAT entry is an enable bit. This enable bit specifies whether the two word pair is to be treated as a valid multi-cast address or not. The bit must be set for the address to be considered valid. The enable bit is not considered when sorting the addresses within the table in order to present them in numerical order.

If the port is enabled after initialization without loading the MCAT, no multi-cast addresses are enabled. The only packets that will pass the receive address filter will be those that match the physical address of the port, and BROADCAST (destination address of all ones) packets.

6.5 Received Packet Filtering

For a packet to be processed by the port it must pass the receive address filter. This filtering occurs according to mode for every packet received over the NI wire. Packets which do not pass the address filter are discarded without increment of a Datagram Discarded event counter; by definition packets which fail to pass the address filter are not addressed to the receiving port.

Received packet filetering only occurs for packets intercepted by the NI link while the microcode is in the ENABLED state.

If a received packet has an address of BROADCAST (destination field all ones,) then the packet passes the receive address filter, and is processed.

If PROMISCUOUS mode is set, all packets received by the port are considered addressed to it, and are processed. Otherwise...

If ALL MULTI-CAST mode is set, all received packets with the multi-cast address bit in the destination address field set will pass the address filter. This mode is a promiscuous mode for multi-cast packets only. In addition under this mode, if a received packet's destination matches the physical address of the port, the packet passes the receive address filter. Otherwise...

A packet passes the filter if its destination address matches the port physical address, or if the destination address is a multi-cast address, and matches a multi-cast address enabled in the multi-cast address table.

6.6 Port Commands And Responses

Commands and Responses are the packets that the driver and the port send to each other in order to communicate. A Command is a request from the port driver for the port to provide some service. The placing of a command on the command queue triggers processing in the NI port. "Placing a command on the queue" includes setting the command queue available bit in the port status register if the command is linked onto an empty command queue.

A response is a packet from the port to the driver informing it of an event. This event may be the reception of an incoming packet, the completion of a command, or an error which occurred while processing a command.

6.6.1 Command Processing -

When commands are present upon the command queue, the port microcode will delink the first command on the command queue, process the command, and build a response if either an error occurred or if the port driver specifically requested a response packet by setting the Response bit (R bit) in the FLAGS field.

Command processing commences when the port discovers it has a packet linked on the command queue (see PCB definition.) The port always checks after processing a command to see if another exists on the command queue. If not, the port goes idle. The port wakes up again after the NI driver has placed a command upon the command queue, and writes the Command Queue Available bit in the CSR Register to indicate that a new command is available.

6.6.2 Responses -

Responses to commands are built if either an error occurred while processing a command, or if the RESP bit is set in the FLAGS field of the original command packet. If a response is not built, then the command entry is linked onto the end of one of the various free queues, depending upon the protocol type of the message, if the command is a Send Datagram, and onto the Unknown Protocol Type free queue if the command does not generate a packet.

When a response is built, it will be linked onto the end of the Response queue anchored in the PCB. The driver is to remove these entries by delinking them from the head of the queue, and processing them. If the response queue is empty when a new entry is added by the port, then a host interrupt is submitted, if interrupts are enabled.

6.6.3 Remote Responses -

Packets which are received over the NI wire also generate response packets, which are linked onto the end of the response queue, as above. Received datagrams are marked as such by an opcode of 5. There is no LSD type processing for incoming datagrams.

Remote responses are built in queue entries gotten from one or the other of the free queues. If the Protocol Type of the received datagram is found as an enabled entry in the Protocol Type Table, then the free queue entry is obtained from the free entry list pointed to by the queue header pointer of the PTT entry. If the Protocol Type is not found in the PTT, then the free entry is obtained from the Unknown Protocol Type free queue, anchored in the PCB. If the queue examined in order to get the free entry is empty, then the received datagram is DISCARDED, and the Datagram Discarded event counter for that queue is incremented.

Note that processing of a received packet takes priority over processing a command packet. Processing of both command packets and received packets are atomic operations, and will not interrupt each other.

6.6.4 Self Directed Commands -

It is specifically allowed for a datagram transmitted to be destined for the transmitting node. The design of the NI Adapter (the NI physical channel), however, shares the CRC generator/checker between the receive and transmit circuits. Thus, since the CRC circuits must be used to check the incoming packet, packets destined for the transmitting node must supply a CRC code that will be transmitted in place of the internally generated CRC. This driver CRC is appended onto the end of the normal data packet, and the ICRC (Included CRC) bit of the packet FLAGS field is set. The text length does not include the appended CRC, which must be generated according the algorithm specified in the Ethernet bluebook specification. This CRC is four bytes in length.

6.6.5 Port Functions -

The following functions are available to the port driver:

1. SNDDG - Send Datagram - Causes a NI datagram to be built and transmitted as an Ethernet packet.

2. LDPTT - Load Protocol Type Table - Causes the Protocol Type Table (PTT) to be internally cached by the port.

3. LDMCAT - Load Multi-cast Address Table - Causes the Multi-cast Address Table (MCAT) to be internally cached by the port.

4. RCCNT - Read and Clear Counters - Causes the event counters kept by the port microcode to be cleared according to the FLAGS bit CLRCTR supplied in the command packet. If the Response (RESP) bit in the FLAGS field is set, a response is generated containing the event counters kept by the port microcode. This command is a performance monitoring and diagnostic feature.

5. WRTPLI - Write PLI- Causes a PLI write with the specified function and data byte. This command is a diagnostic feature.

6. RDPLI - Read PLI - Causes a PLI read with the specified function. If a response is built for this command, the data byte read from the PLI is returned. This command is a diagnostic feature.

7. RDNSA - Read NI Station Address - Reads the NI station address from the NI link physical address ROMs. This command also reports the state of several link mode bits.

8. WRTNSA - Write NI Station Address - Writes the NI station into the NI link address RAMs. Also sets the state of several link mode bits.

6.7 Error Handling

The port microcode is capable of retrying many operations that fail; this allows a large part of the error recovery to be built into the port microcode, and removed from the port driver. When the port encounters a fatal, non-recoverable error, the port will stop processing any more commands.

The error handling procedures for the NI20 are specified in the KLNI Error Specification, written by Joe Holewa.

6.7.1 Port Hardware Errors -

In addition, there are a class of more severe errors which cause the port to cease operations immediately, and to exit to a special microcode location that has a CRAM parity error. The console will notice the error location, and will re-initialize the port, if possible. The errors that result in this action are listed below:

1. Loc. 7760 -INTERNAL.ERROR -internal consistancy error

2. Loc. 7761 -FAIL.SELFTEST -start up self test failed

3. Loc. 7762 -FAIL.EBUS -ebus parity error

4. Loc. 7763 -FAIL.PLIPE -unrecoverable PLI parity error

5. Loc. 7764 -FAIL.CHANNEL -fatal channel error

6. Loc. 7765 -FAIL.CBUSPE -unrecoverable cbus parity error

7. Loc. 7766 -FAIL.XMIT.AT -spurrious transmitter attention

8. Loc. 7767 -FAIL.RCV.AT -spurrious receiver attention

9. Loc. 7770 -FAIL.XMT.TIM -transmitter timeout

10. Loc. 7771 -FAIL.PARPRE -unrecoverable formatter parity predictor error

11. Loc. 7772 -USED.BUFFER.PE -used buffer list parity error

12. Loc. 7773 -FREE.BUFFER.PE -free buffer list parity error

13. Loc. 7774 -XMIT.BUFFER.PE -unrecoverable transmit buffer parity error

6.7.2 Error Events -

Various errors may occur which do not necessarily imply a hardware malfunction. These errors include excessive collisions, CRC errors, framing errors, etc. The occurrence of these errors terminates the packet transmission or reception in progress when the error occurs, but do not effect the processing of other commands or packets being received. Errors of this sort are reported by creating a response packet for the transfer involved, and setting the STATUS field accordingly. A list of possible error conditions is provided below:

1. Unrecognized command - Packet has invalid operation code.

2. Buffer length violation - BSD has inconsistent buffer length information.

3. CRC error - Received packet has CRC error.

4. Framing error - Received data not byte aligned.

5. Packet too long - Packet length exceeds maximum Ethernet packet length.

6. Excessive Collisions - Packet collided 16 times in succession.

7. Carrier Lost - Carrier lost before end of packet detected.

8. Collision detect check - Collision detect heartbeat failed to assert.

6.7.3 Datagram Discarded -

Under certain conditions, received datagrams may be discarded due to insufficient buffer space. This is a characteristic of the datagram class service that is offered by the NI port. To wit, if a received datagram is occupying buffer space in the NIA, and obtaining a free queue entry from some free queue fails, then the datagram involved is discarded, and the Datagrams Discarded counter for the appropriate free queue is incremented. This occurs even if buffer space in the link is still available. The rationale for this is that 1) other packets being received may be able to be stored since the free queue error condition is protocol type dependent, and 2) if the link fills up, datagrams will be discarded without increment of a Datagrams Discarded event counter. No other error indication or response is made. It is up to higher levels of the network to detect and recover from such errors.

If the port is addressed by a burst of packets such that the internal buffer space in the NI physical channel is exhausted, datagrams may be discarded without increment of the Datagrams Discarded counter. The buffer space allowed in the physical channel (NI link) is sufficient to make this a low probability event. The NIA has 16K bytes of buffering organized as 32 X 512 byte buffers. This is sufficient to store thirty two (32) minimum sized packets and ten (10) maximum sized packets.

6.8 Event Counters

In order to provide for performance measurement, and diagnostic purposes, a number of event counters are provided by the port. These record the occurrence of certain events, and can be read and cleared upon command by the driver program. These event counters do not necessarily record, or imply, abnormal errors, although high counting rates for some of them may indeed point to a hardware or software failure. A list of the applicable events is given below:

1. Received a byte [1]

2. Transmited a byte [1]

3. Received a frame

4. Transmited a frame

5. Received a multi-cast byte [1]

6. Received a multi-cast frame

7. Transmited a frame that was initially defered

8. Transmited a frame with a single collision

9. Transmited a frame with multiple collisions

10. Failed to successfully transmit a frame

11. Send failure reason bit mask [2]

12. Transmited a frame with a late collision

13. Collision detect check failed to assert

14. Failed to successfully receive a frame

15. Received failure reason bit mask [2]

16. Discarded a datagram for Unknown Protocol Type free queue

17. Discarded a datagram for Protocol Type entry 1 free queue

18. Discarded a datagram for Protocol Type entry 2 free queue

19. Discarded a datagram for Protocol Type entry 3 free queue

20. Discarded a datagram for Protocol Type entry 4 free queue

21. Discarded a datagram for Protocol Type entry 5 free queue

22. Discarded a datagram for Protocol Type entry 6 free queue

23. Discarded a datagram for Protocol Type entry 7 free queue

24. Discarded a datagram for Protocol Type entry 8 free queue

25. Discarded a datagram for Protocol Type entry 9 free queue

26. Discarded a datagram for Protocol Type entry 10 free queue

27. Discarded a datagram for Protocol Type entry 11 free queue

28. Discarded a datagram for Protocol Type entry 12 free queue

29. Discarded a datagram for Protocol Type entry 13 free queue

30. Discarded a datagram for Protocol Type entry 14 free queue

31. Discarded a datagram for Protocol Type entry 15 free queue

32. Discarded a datagram for Protocol Type entry N free queue

[1] Although this counter is indicated as counting when a byte is transfered, in the interest of efficiency the total of bytes transfered in a packet transmit or receive is added to the appropriate counter after the transfer is complete. Listing these event counters in the fashion indicated is a good abstraction.

[2] This bit mask provides an indication of which errors have occured, but does not indicate how many of which errors have occured. The bit mask is defined in the section describing the Read Counters command.

The Read and Clear Performance Counters command reads the values of these counters and returns them in a response packet.

6.9 MOP Message Handling

It is required by the Network Managment specification that each NI node handle certain Maintenance Operation Protocol messages, in particular, Request ID (REQID), Loopback (LPBK), and Read Counters (RDCNT.) In addition, the Ethernet blue book specifies these client layer protocols. Due to microcode space considerations, these features are not implemented in the NI20 port microcode.

6.10 Broadcast Of System ID

It is required by the NI product architecture for each NI node to periodically transmit a system ID message on the system configuration multicast address. The port will not perform this function. It is the responsibility of the port driver and operating system to provide this feature.

6.11 Port States

There are four LCG NI port states. These states are listed below.

1. Disabled - The port is offline; Local commands will be processed.

2. Initialize - The port is coming online; functionality not yet available.

3. Enabled - The port is online; full functionality is available.

4. Halted - The port microsequencer is halted; nothing is available.

These four states form a complete description of the port state.

7.0 COMMAND SPECIFICATION

The following sections discuss the format of the various commands and responses that the LCG NI port recognizes, and can produce.

7.1 FLAGS Field

The format of the FLAGS field is show below: Note that there are two formats; one for the Send Datagram command and response (and the Datagram Received response,) and one for all other commands and responses.

Send Datagram

0	1	2	3	4	5	6	7
Pack	ICRC	PAD	RSVD	BSD	RSVD	RSVD	Resp

Not Send Datagram

RSVD	RSVD	RSVD	RSVD	RSVD	RSVD	CLRCTR	Resp

Pack

Packing Format - Defines the packing format for non-BSD datagrams. If this bit is 0, the datagram is in industry compatible mode. If this bit is 1, the datagram is in high density mode. All datagrams are always transmitted left-to-right.

BSD

Buffer Segment Descriptor - If this bit is 1, the datagram is using the BSD format described under the Send Datagram command. If the bit is 0, the datagram is in 'immediate' mode; that is, the data to be transmitted follows the destination address in the queue entries.

ICRC

When set, the port driver has appended a four character CRC at the end of the datagram. This CRC is to be used instead of the internally generated CRC. This feature must be used in the transmission of self-directed datagrams. The length fields in the command packets do NOT reflect the addition of the four extra characters.

PAD

When set, the port will pad packets less than the Ethernet minimum size by appending 44 bytes to the end of the packet. In addition, two bytes indicating the valid data length will be prepended to the text data. These length bytes are transmitted low order byte first, and indicate the number of actual text bytes (not including padding) that occur within the packet. The two length bytes are always transmitted when padding is enabled, regardless of whether padding was necessary for a particular packet transmitted.

When clear, the port does not do such padding. If a packet is padded, it will be padded with zeros. If a packet is presented to the port without this bit set, and it is less than 46 bytes, an error response for a "runt" packet is generated. Packets presented that are larger than the maximum Ethernet packet size always cause a length error, and generate an error response packet.

Resp

Response - When this bit is 1, the port will always build a response after processing the command.

CLRCTR

This field is valid only for the Read Counters command. If this bit is set, all the counters will be cleared after their values are reported in the response packet for this command.

7.2 Status Field

The following field is used by the port to report the status of a completed command. This field appears in the command opcode word of the queue entry. When valid, this field indicates the logging of an exception event.

```
    0       1       2       3       4       5       6       7
+-------+-------+-------+-------+-------+-------+-------+-------+
| Spare | Send/ |              Error Type               | Error |
|       |Receive|       |       |       |       |       |       |
+-------+-------+-------+-------+-------+-------+-------+-------+
```

Send / Receive

When this bit is zero, an error occurred on receive; receive failed. When this bit is one, an error occurred on transmit; transmission failed.

Error Type

This field indicates the error event being logged. The field is set according to the following table:

Bit value (octal)	Event type
00	Excessive collisions [2]
01	Carrier check failed (carrier lost) [2]
02	Collision detect check failed
03	Short circuit [4]
04	Open circuit [4]
05	Frame too long
06	Remote failure to defer (late collision)
07	Block check error (CRC error)
10	Framing error
11	Data overrun (NIA buffer space exhausted)
12	Unrecognized protocol type
13	Frame too short [3]
31	Queue length violation
32	Illegal PLI function
33	Unrecognized command
34	Buffer length violation [1]
35	Free buffer list parity error [5]
36	Transmit buffer parity error

[1] For a transmission, this error means that the length information in the transmitted BSD was inconsistent. One such case is when the length field of the transmitted datagram does not match the total length of the BSDs to be transmitted.

[2] When this event is being logged, bits 26-35 of the opcode word in the queue entry become the TDR reference number obtained from the NIA when the error occurred. This indicates the time, in 100 nsec tics, from the time the transmission started until the error event was detected by the NIA hardware.

[3] This error occurs only when padding of the transmitted frame is disabled, and the frame length is smaller than the smallest legal Ethernet frame size of 46 data bytes (64 bytes including all physical channel protocol.) No indication is given if the frame size would be runt, and padding is enabled. In that case, the frame is padded as noted in the description of the PAD flag.

[4] These errors are not detectable using the NIA module. These error bits will never be set. They are included to conform with the Network Management functional specification.

[5] This is a hardware error; it is reported here due to unusual action on the part of the port hardware. This error is recovered by doing a complete, asynchronous initialization of the NIA. Any transmission in progress is lost. The receive in progress is lost. Any packets stored in the NIA buffers are lost. No increment of datagram discarded counters is made. This error cannot be recovered. Port operation continues, however. If the driver sees many of these, it should shut the port down.

Error

When this bit is zero, the status field has no meaning; and Must Be Zero. When this bit is one, the status field is reporting an error event; the above definition of the error type fields, and of the direction field comes into effect.

7.3 Send Datagram

Send Datagram causes an NI datagram to be built and sent to the destination port address.

7.3.1 SNDDG Command -

The queue entry must be allocated on a four word boundary. Data bytes are always transmitted from left to right. That is, for a given word, the data in bits 0-7 is transmitted first, the data in bits 8-15 is transmitted second, the data in bits 16-24 is transmitted third, etc. This left to right format is the norm for all data transmitted.

Note that chains of BSDs and text length fields must always describe full bytes. It is illegal for a buffer to terminate in the middle of a byte. The result if this restriction is violated is undefined; the system may fail.

A byte may be split accross two BSDs if both are in high density mode, and the first BSD terminates in a word boundary which is the end of the first word of a two word pair.

The format of this command as a command queue entry is specified below:

SNDDG Command Format (BSDs)

```
+---------------------------------------------------------+
|                       Queue FLINK                       |
+---------------------------------------------------------+
|                       Queue BLINK                       |
+---------------------------------------------------------+
|                  Reserved for Software                  |
+---------------------------------------------------------+
|  <0-7>   |  <8-15>  |   <16-23>   |                     |
| Status   |  Flags   |   Opcode    |         MBZ         |
+---------------------------------------------------------+
|              <0-19>              |       <20-35>        |
|               MBZ                | Length of text data  |
+---------------------------------------------------------+
|                       |        <16-31>        |  |
|          MBZ          |  Protocol Type Value  |  |
+---------------------------------------------------------+
|                   FREEQ header address                  |
+---------------------------------------------------------+
|                 High order Destination                  |
+---------------------------------------------------------+
|                  Low order Destination                  |
+---------------------------------------------------------+
|                    BSD base address                     |
+---------------------------------------------------------+
                             .
                             .
                             .
+---------------------------------------------------------+
|                       *Queue End                        |
+---------------------------------------------------------+
```

Note that although the BSD may be allocated within the queue entry that the BSD header must still be allocated on a four word boundary. From the diagram it is clear that the BSD base address is not allocated at the end of a four word block. Thus, the driver cannot start the BSD until the first four word boundary following the BSD base address pointer. Also remember that this BSD base pointer is a physical address.

SNDDG Command Format (non-BSD)

```
+-----------------------------------------------------------+
|                       Queue FLINK                         |
+-----------------------------------------------------------+
|                       Queue BLINK                         |
+-----------------------------------------------------------+
|                  Reserved for Software                    |
+-----------------------------------------------------------+
|  <0-7>     |  <8-15>   |   <16-23>    |                   |
| Status     |  Flags    |   Opcode     |        MBZ        |
+-----------------------------------------------------------+
|             <0-19>                   |      <20-35>       |
|              MBZ                     | Length of text data|
+-----------------------------------------------------------+
|                            |       <16-31>          |     |
|          MBZ               | Protocol Type Value    |     |
+-----------------------------------------------------------+
|                   FREEQ header address                    |
+-----------------------------------------------------------+
|                 High order Destination                    |
+-----------------------------------------------------------+
|                  Low order Destination                    |
+-----------------------------------------------------------+
|                       Text Data 0                         |
+-----------------------------------------------------------+
|                       Text Data 1                         |
+-----------------------------------------------------------+
|                       Text Data 2                         |
+-----------------------------------------------------------+
                             .
                             .
                             .
+-----------------------------------------------------------+
|                      Text Data N-1                        |
+-----------------------------------------------------------+
|                       Text Data N                         |
+-----------------------------------------------------------+
|                       *Queue End                          |
+-----------------------------------------------------------+
```

When this command is issued, the port will build and send a datagram to the port designated.

Opcode

For a Send Datagram command, the opcode field is a 1.

High / Low Destination

These two words specify the destination address of the packet. Their format is described below: Note that byte 0 is the first byte transmitted on the NI, and byte 5 is the last byte transmitted. Bit 31 of the high order destination is therefore the Multicast address bit. This ordering is compatible with the ordering for VAX NIs.

Destination Address Format

```
0.........7 8.........15 16........23 24........31   32-35
+---------------------------------------------------------+
| Byte 3    |    Byte 2   |  Byte 1    |  Byte 0     | MBZ |   Low
+---------------------------------------------------------+
                                                    ^
                                          multicast bit

0.....................15 16........23 24........31   32-35
+---------------------------------------------------------+
|           MBZ          |    Byte 5  |  Byte 4    | MBZ |   High
+---------------------------------------------------------+
```

7.3.2 DGSNT Response -

A response to the SNDDG command will be built if 1) an error occurred during processing of the command, or 2) the Resp bit of the FLAGS field of the original datagram packet was set. The format of the returned packet is shown below. Note that the format of the command follows the format of the original response; i.e., if the BSD bit of the FLAGS field was on, then the response will be in BSD format as well. The format will be as follows for a Non-BSD packet:

DGSNT Response Format (Non-BSD)

```
+-------------------------------------------------------+
|                     Queue FLINK                       |
+-------------------------------------------------------+
|                     Queue BLINK                       |
+-------------------------------------------------------+
|                Reserved for Software                  |
+-------------------------------------------------------+
|  <0-7>   |  <8-15>  |   <16-23>   |                   |
| Status   |  Flags   |   Opcode    |        MBZ        |
+-------------------------------------------------------+
|             <0-19>              |       <20-35>       |
|              MBZ                | Length of text data |
+-------------------------------------------------------+
|                            |      <16-31>        |    |
|           MBZ              | Protocol Type Value |    |
+-------------------------------------------------------+
|                 FREEQ header address                  |
+-------------------------------------------------------+
|                High order Destination                 |
+-------------------------------------------------------+
|                Low order Destination                  |
+-------------------------------------------------------+
|                     Text Data 0                       |
+-------------------------------------------------------+
|                     Text Data 1                       |
+-------------------------------------------------------+
|                     Text Data 2                       |
+-------------------------------------------------------+
                           .
                           .
                           .
+-------------------------------------------------------+
|                    Text Data N-1                      |
+-------------------------------------------------------+
|                     Text Data N                       |
+-------------------------------------------------------+
```

The format of a response for a send datagram command packet with the BSD bit of the FLAGS field set is shown below:

LCSNT Response Format (BSD)

```
+----------------------------------------------------------+
|                      Queue FLINK                          |
+----------------------------------------------------------+
|                      Queue BLINK                          |
+----------------------------------------------------------+
|                 Reserved for Software                     |
+----------------------------------------------------------+
|  <0-7>   |  <8-15>  |  <16-23>   |                        |
| Status   |  Flags   |  Opcode    |          MBZ           |
+----------------------------------------------------------+
|                <0-19>              |      <20-35>         |
|                 MBZ                | Length of text data  |
+----------------------------------------------------------+
|                             |        <16-31>         |   |
|            MBZ              |  Protocol Type Value   |   |
+----------------------------------------------------------+
|                  FREEQ header address                     |
+----------------------------------------------------------+
|                 High order Destination                    |
+----------------------------------------------------------+
|                  Low order Destination                    |
+----------------------------------------------------------+
|                    BSD base address                       |
+----------------------------------------------------------+
                             .
                             .
                             .
+----------------------------------------------------------+
|                        Text N                             |
+----------------------------------------------------------+
```

The format of the individual words and the bits therein conform to the information given for the Send Datagram command format, above.

7.3.3 DGRCV Response -

When a datagram is received, a response packet is built. The format of a received datagram is as follows. Note that a received datagram is never received as a BSD packet.

DGRCV Response Format

```
   +--------------------------------------------------------+
   |                      Queue FLINK                       |
   +--------------------------------------------------------+
   |                      Queue BLINK                       |
   +--------------------------------------------------------+
   |                 Reserved for Software                  |
   +--------------------------------------------------------+
   | <0-7>    | <8-15>   |  <16-23>    |                    |
   | Status   | Flags    |  Opcode     |        MBZ         |
   +--------------------------------------------------------+
   |               <0-19>              |       <20-35>       |
   |                MBZ                | Length of text data |
   +--------------------------------------------------------+
   |                 High order Destination                 |
   +--------------------------------------------------------+
   |                 Low order Destination                  |
   +--------------------------------------------------------+
   |                   High order Source                    |
   +--------------------------------------------------------+
   |                    Low order Source                    |
   +--------------------------------------------------------+
   |                             |       <16-31>         |  |
   |            MBZ              | Protocol Type Value   |  |
   +--------------------------------------------------------+
+--|                      Data Pointer                      |
|  +--------------------------------------------------------+
|
|
|
|
|  +--------------------------------------------------------+
+->|                      Text Data 0                       |
   +--------------------------------------------------------+
   |                      Text Data 1                       |
   +--------------------------------------------------------+
   |                      Text Data 2                       |
   +--------------------------------------------------------+
                               .
                               .
                               .
   +--------------------------------------------------------+
   |                     Text Data N-1                      |
   +--------------------------------------------------------+
   |                      Text Data N                       |
   +--------------------------------------------------------+
```

Opcode

The Opcode field for a received datagram is 5.

Length of Text Data

This field contains the length of the transfered text data in bytes, plus four (4) to include the cyclic redundancy check bytes at the end of the packet.The length value does not include the packet header. This is the length of the data portion of the datagram. The packet length field always indicates the actual number of bytes in the packet, even if the packet is too long, or in an error of some sort occurs. Note that the received CRC bytes are placed into the host memory buffer immediately following the end of memory data. This is to allow a software double check of packet integrity. If the packet is too large for the receiving buffer, data is delivered until the end of the buffer is reached. No CRC is appended to the data in this case. In no case will the data transfered to memory (including CRC) exceed the boundary of the buffer.

Protocol Type

This field contains the protocol type of the received packet. The format for this field is the same as for the send datagram command, and as for the protocol type field of an PTT entry. Note that the protocol type free queue header address is omitted from the received packet. In this case, the information is superfluous.

Destination High/Low

This field contains the destination port address as received from the NI wire. This field is included so that messages received for a multicast address can be distinguished from messages received for the physical address of the port.

Source High/Low

This field contains the originating port address. The format of this address is given below. In the following diagram, byte 0 is the first byte received over the NI wire. Byte 5 is the last byte received. Thus, bit 31 is the multicast address bit.

Port Address Format

```
0.........7 8..........15 16........23 24........31  32-35
+--------------------------------------------------------+
| Byte 3    |   Byte 2    |  Byte 1    |  Byte 0    |  MBZ |  Low
+--------------------------------------------------------+
                                                   ^
                                          multicast bit

0....................15 16........23 24........31  32-35
+--------------------------------------------------------+
|         MBZ           |     Byte 5   |  Byte 4    |  MBZ |  High
+--------------------------------------------------------+
```

Data Pointer

This word contains the physical address of the area where data is stored. The text area pointed to is assumed word aligned, and in industry compatible mode.

Text Data

Text Data 0 contains the first byte of the received packet. The text portion of a received datagram is always stored left to right. That is, the first byte of text received will be placed into bits 0-7 of the first text word, the second byte of text received will be placed into bits 8-15, the third into bits 16-23, the fourth into bits 24-31, etc. This occurs for all data modes.

7.4 Load Protocol Type Table

When this command is issued, the PTT specified will be cached internally to the port. Note that the protocol free queue addresses are cached internally as well; this means that the free queue headers cannot change addresses unless this command is issued as well. Note that the addresses specified in this table point to the free queue header, not to the first queue entry of the queue chain.

After port initialization, no protocol types are enabled. All frames which pass receive address filtering will be delivered to the unknown Protocol Type queue.

The address of the PTT specified in the PCB may not be altered in a running port without doing an initialization.

In order to add or delete a new protocol type to or from a running port, a new protocol type table is built over the old at the address specified by the driver in the PCB pointer to the PTT, including the new protocol types, and omitting any which are to be deleted. As before, the table must be built with the protocols in ascending order. This applies even to entries which are disabled when the table is loaded. The port sets the enable bits of the protocol types that are to be enabled, and issues the LDPTT command. When the PTTLD response is received by the driver, the new protocol types are in effect.

Since no responses to received datagrams will be started while the load of this table is in progress, there is no danger of compromising the queue structures.

In order to enable or disable a given protocol type without removing or adding protocols from the PTT, the port need only set the enable bits in the protocol type table as appropriate, and issue a LDPTT command as before.

7.4.1 LDPTT Command - The format of this command is specified below:

LDPTT Command Format

```
+-----------------------------------------------------------+
|                        Queue FLINK                        |
+-----------------------------------------------------------+
|                        Queue BLINK                        |
+-----------------------------------------------------------+
|                   Reserved for Software                   |
+-----------------------------------------------------------+
|  <0-7>    |  <8-15>   |   <16-23>   |                     |
|  Status   |  Flags    |   Opcode    |         MBZ         |
+-----------------------------------------------------------+
```

Opcode

The opcode for this command packet is 3. When the command is recognized, the Protocol Type Table is internally cached. There is no additional queue data.

If the LDPTT command is not executed before enabling the port after initialization, then no protocol types are enabled. Any packets that pass the receive address filter will be linked onto the Unknown Protocol Type queue.

7.4.2 PTTLD Response -

The Load Protocol Type Table command signals completion by building a response, and putting that response onto the end of the response queue. The format of that response is given below:

PTTLD Response Format

```
+-------------------------------------------------------------+
|                        Queue FLINK                          |
+-------------------------------------------------------------+
|                        Queue BLINK                          |
+-------------------------------------------------------------+
|                    Reserved for Software                    |
+-------------------------------------------------------------+
|  <0-7>     |  <8-15>   |   <16-23>    |                     |
|  Status    |  Flags    |   Opcode     |         MBZ         |
+-------------------------------------------------------------+
```

The driver should not assume that a protocol type has been enabled until this response is received from a Load PTT command. The Opcode and flags are not modified by this command.

7.5 Load Multi-cast Address Table

When this command is issued, the Multi-cast address table (MCAT) is loaded from the table whose address is specified by the PCB. At initialization time, no multi cast addresses are enabled.

In order to add or delete a multi-cast address to or from a running port, a MCAT is built in memory at the address pointed to by the PCB MCAT base address pointer with the addresses in increasing numerical order. Then the LDMCAT command is issued. When the response to this command is returned the multi-cast addresses specified in this table are in effect.

7.5.1 LDMCAT Command -

The table is loaded into the internal cache of the port. This table must follow the format stated for the MCAT. This table must be loaded before address checking will take effect.

LDMCAT Command Format

```
+-------------------------------------------------------------+
|                        Queue FLINK                          |
+-------------------------------------------------------------+
|                        Queue BLINK                          |
+-------------------------------------------------------------+
|                   Reserved for Software                     |
+-------------------------------------------------------------+
|  <0-7>    |  <8-15>  |   <16-23>   |                        |
| Status    |  Flags   |   Opcode    |          MBZ           |
+-------------------------------------------------------------+
```

Opcode

The operation code for this command is 2. There is no other queue data.

If the multi-cast address table is not loaded before enabling the port after initialization, then no multi-cast address are enabled. The receive address filter will reject all multi-cast packets, except for those noted above.

7.5.2 MCATLD Response -

The successful completion of the LDMCAT command is signaled by the building of the following response on the response queue. The format of the response is illustrated below: Note that the driver can assume that the specified Multi-cast address table has been loaded only after it has received this response packet.

MCATLD Response Format

```
+-----------------------------------------------------------+
|                        Queue FLINK                        |
+-----------------------------------------------------------+
|                        Queue BLINK                        |
+-----------------------------------------------------------+
|                  Reserved for Software                    |
+-----------------------------------------------------------+
|  <0-7>    |  <8-15>  |   <16-23>    |                     |
| Status    |  Flags   |   Opcode     |         MBZ         |
+-----------------------------------------------------------+
```

Note that the response is built only if the driver sets the RESP bit of the flags field in the original command packet. The command opcode and flags are not modified by this command.

7.6 Read And Clear Performance Counters

This command will read the performance counters, returning their value in the CNTRC response, and clear the performance event counters as specified by the CLCNT bit in the flags byte of the command.

7.6.1 RCCNT Command -

The format of the command queue entry to accomplish this is given below:

RCCNT Command Format

```
+-----------------------------------------------------------+
|                        Queue FLINK                        |
+-----------------------------------------------------------+
|                        Queue BLINK                        |
+-----------------------------------------------------------+
|                  Reserved for Software                    |
+-----------------------------------------------------------+
|  <0-7>    |<8-13> <14>    <15>| <16-23> |                 |
| Status    | Flags CLRCTR      | Opcode  |       MBZ       |
+-----------------------------------------------------------+
```

Opcode

The opcode for this command packet is 4. This command does not generate an NI packet. A response is built if the RESP bit in the flags word is set.

CLRCTR <14> (Clear Counters)

This bit in the flags byte specifies whether the event counters are to be cleared by the command after the values are read. If this bit is a 1, then the counters will be cleared.

The Datagrams discarded counters are kept for each of the protocol type queues, and for the unknown protocol type queue. Thus, the driver can get an indication when a protocol type free queue is exhausted by execution of this command. A counter is returned for every protocol type entry allowed in the PTT. Those counters which correspond with protocol type entries which are not enabled are returned with a value of zero.

7.6.2 CNTRC Response -

If the RESP of the FLAGS field of the original command packet is on, then a response to the above command is built. The response has the following format:

CNTCL Response Format

```
+----------------------------------------------------------+
|                       Queue FLINK                        |
+----------------------------------------------------------+
|                       Queue BLINK                        |
+----------------------------------------------------------+
|                  Reserved for Software                   |
+----------------------------------------------------------+
|   <0-7>    |   <8-15>   |   <16-23>    |                 |
|   Status   |   Flags    |   Opcode     |       MBZ       |
+----------------------------------------------------------+
|                           BR                             |
+----------------------------------------------------------+
|                           BX                             |
+----------------------------------------------------------+
|                           FR                             |
+----------------------------------------------------------+
|                           FX                             |
+----------------------------------------------------------+
|                          MCBR                            |
+----------------------------------------------------------+
|                          MCFR                          |
+--------------------------------------------------------+
|                          FXID                          |
+--------------------------------------------------------+
|                          FXSC                          |
+--------------------------------------------------------+
|                          FXMC                          |
+--------------------------------------------------------+
|                           XF                           |
+--------------------------------------------------------+
|                          XFBM                          |
+--------------------------------------------------------+
|                          CDCF                          |
+--------------------------------------------------------+
|                           RF                           |
+--------------------------------------------------------+
|                          RFBM                          |
+--------------------------------------------------------+
|                         DDUPT                          |
+--------------------------------------------------------+
|                         DDPT1                          |
+--------------------------------------------------------+
|                         DDPT2                          |
+--------------------------------------------------------+
|                         DDPT3                          |
+--------------------------------------------------------+
|                         DDPT4                          |
+--------------------------------------------------------+
|                         DDPT5                          |
+--------------------------------------------------------+
|                         DDPT6                          |
+--------------------------------------------------------+
|                         DDPT7                          |
+--------------------------------------------------------+
|                         DDPT8                          |
+--------------------------------------------------------+
|                         DDPT9                          |
+--------------------------------------------------------+
|                        DDPT10                          |
+--------------------------------------------------------+
|                        DDPT11                          |
+--------------------------------------------------------+
|                        DDPT12                          |
+--------------------------------------------------------+
|                        DDPT13                          |
+--------------------------------------------------------+
|                        ......                          |
+--------------------------------------------------------+
|                        DDPTn-1                         |
+--------------------------------------------------------+
|                        DDPTn                           |
+-----------------------------------------------------+
|                        URFD                         |
+-----------------------------------------------------+
|                        DOVR                         |
+-----------------------------------------------------+
|                        SBUA                         |
+-----------------------------------------------------+
|                        UBUA                         |
+-----------------------------------------------------+
|                        RSVD                         |
+-----------------------------------------------------+
```

By combining the read and clear commands, the driver knows that the event counters are not skewed when a clear command is issued. In the following, note that a packet is not "received" unless it passes the receive address filter.

BR
Bytes Received

This counter represents the number of 8 bit data text characters received as datagrams over the NI.

BX
Bytes Transmitted

This counter represents the number of 8 bits data text characters transmitted successfully as datagrams over the NI.

FR
Frames Received

This counter represents the number of frames (packets) that have been received over the NI wire.

FX
Frames Transmitted

This counter represents the number of frames (packets) that have been successfully transmitted over the NI wire.

MCBR
Multi-Cast Bytes Received

This counter represents the number of 8 bit bytes received in packets with the multi-cast bit set in the destination field.

MCFR
Multi-Cast Frames Received

This counter represents the number of frames (packets) that were received with the multi-cast bit in the destination field set.

FXID
Frames Transmitted, Initially Deferred

This counter represents the number of frames (packets) transmitted that had to defer to other traffic on the NI wire before transmission. This event is not detectable using the NIA. Thus, this counter will always appear with a value of zero.

FXSC
Frames Transmitted, Single Collision

This counter represents the number of frames (packets) that were successfully transmitted, and which collided with another transmission exactly once.

FXMC
Frames Transmitted, Multiple Collisions

This counter represents the number of frames (packets) that were successfully transmitted, and which collided with another transmission more than once.

XF
Transmit Failures

This counter represents the number of frames that were not successfully transmitted. This counter is incremented for excessive collisions, parity errors, etc. This counter is associated with the XFBM, which notes occurance of error classes.

XFBM
Transmit Failure Bit Mask

This counter gives the accumulated reasons for transmission failures. The bit meanings are listed below:

1. Bits 0-23 -- Unassigned

2. Bit 24 -- Loss of Carrier

3. Bit 25 -- Transmiter buffer parity error

4. Bit 26 -- Remote failure to defer

5. Bit 27 -- Frame too long

6. Bit 28 -- Open circuit

7. Bit 29 -- Short circuit

8. Bit 30 -- Carrier check failed (collision detect check failed)

9. Bit 31 -- Excessive collisions

CDCF
Collision Detect Check Failed

This counter gives the number of times that the collision detect check failed after a transmit. This is the number of times that heartbeat failed to assert after a transmit ended. This counter has meaning only if the H4000 mode bit is set.

RF
Receive Failures

This counter gives the number of received frames whose reception ultimately failed. This counter is associated with the RFBM counter, which marks occurance of the various error types.

RFBM
Receive Failure Bit Mask

This counter gives the accumulated reasons for receive failures. The bit definitions are given below:

1. Bits 0-26 -- Unassigned

2. Bit 27 -- Free list parity error

3. Bit 28 -- Data overrun (No free buffers)

4. Bit 29 -- Frame too long

5. Bit 30 -- Framing error

6. Bit 31 -- Block check error

DDUPT
Datagram Discarded for Unknown Protocol Type

This counter keeps track of the number of datagrams discarded for the Unknown Protocol Type free queue. Any time a datagram is discarded with an unrecognized protocol type, this counter is incremented.

DDPT1 - DDPTn
Datagram Discarded for Protocol Type N

These counters keep track of the number of datagrams discarded for each of the protocol type free queues. Any time a datagram is discarded due to no available free space, one of these counters is incremented, if the protocol type was recognized. There are as many of these counters as needed to support the number of protocol types allowed in the NI Configuration Register.

URFD
UnRecognized Frame Destination

This counter has no meaning for the NI20, and will always be reported as zero.

DOVR
Data Overrun

This counter represents the number of packets that were received incorrectly due to buffer space in the NIA being exhausted. Such packets are reported in an error response. As much data as was received will be presented to the host. The length field of the datagram received response will represent as much data as was received.

SBUA
System Buffer UnAvailable

This counter has no meaning for the NI20, and will always be reported as zero.

UBUA
User Buffer UnAvailable

This counter represents the total number of packets discarded due to a free queue being exhausted. This number is the sum total of the Datagram Discarded for Protocol Type 'n' counters plus the Datagram Discarded for Unknown Protocol Type counter.

RSVD
Reserved for Microcode

This counter is reserved for the port microcode and presently it will be returned as zeroes.

7.6.2.1 WRTPLI Command -

Note that execution of illegal or random commands will compromise the functionality of the interface in an indeterminate fashion. It is recommended that the port not be executing meaningful commands when this command is issued. This command is for diagnostic purposes only.

The command has the following format:

WRTPLI Command Format

```
+-------------------------------------------------------------+
|                         Queue FLINK                         |
+-------------------------------------------------------------+
|                         Queue BLINK                         |
+-------------------------------------------------------------+
|                    Reserved for Software                    |
+-------------------------------------------------------------+
|  <0-7>    |  <8-15>  |   <16-23>    |                       |
|  Status   |  Flags   |   Opcode     |          MBZ          |
+-------------------------------------------------------------+
|                 | <20-23>  |              | <28-35>         |
|      MBZ        | Control  |     MBZ      | PLI data        |
+-------------------------------------------------------------+
```

Opcode

The operation code for this packet is 6. If the response bit in the Flags word is on, then a response confirming the correct execution of this command will be built and placed onto the response queue anchored in the PCB. This command, along with RDPLI, gives the port driver the same visibility into the Port Link Interface that the port itself has.

Control

This field specifies the PLI command which is to be put onto the PLI when the write cycle is done. The command codes map into PLI commands as defined in the following table:

Value (octal)	Function
0	- Illegal
1	- SEL.TO.XMIT.BUF
2	- WR.FREE.LIST
3	- SEL.REL.BUF
4	- WR.XMIT.ACTION
5	- RES.REL.ATTN
6	- ENA.LINK.CTL
7	- DIS.LINK.CTL
10	- WR.PREP
11	- WR.XMIT.BUF
12	- WR.REG

For more information as to what these commands do functionally, please consult the NIA hardware specification. Any function attempted which is not defined by this table will return with the status field indicating an Illegal PLI function.

PLI data

This field specifies the PLI data bits to be placed onto the PLI when the write cycle is done. This field is eight bits wide.

7.6.2.2 PLIWRT Response -

The write PLI command results in a response if the RESP bit of the FLAGS field of the original packet was set. If built, the response has the following format:

PLIWRT Response Format

```
+----------------------------------------------------------+
|                        Queue FLINK                       |
+----------------------------------------------------------+
|                        Queue BLINK                       |
+----------------------------------------------------------+
|                   Reserved for Software                  |
+----------------------------------------------------------+
|  <0-7>   |  <8-15>  |   <16-23>   |                      |
| Status   |  Flags   |   Opcode    |         MBZ          |
+----------------------------------------------------------+
|                 | <20-23>  |                | <28-35>  |
|      MBZ        | Control  |      MBZ       | PLI data |
+----------------------------------------------------------+
```

The driver is assured that the port interface has executed this command only after the response has been received.

7.6.3 Read Port/Link Interface -

When executed, this command causes a READ PLI cycle to be executed, using the control bits specified. The data is returned in the response queue entry.

7.6.3.1 RDPLI Command -

The data for this command is returned in the response entry built if the RESP flag is set when the command is executed.

The format of this command is specified below:

RDPLI Command Format

```
+-----------------------------------------------------------+
|                        Queue FLINK                        |
+-----------------------------------------------------------+
|                        Queue BLINK                        |
+-----------------------------------------------------------+
|                  Reserved for Software                    |
+-----------------------------------------------------------+
|  <0-7>    |  <8-15>   |   <16-23>    |                    |
| Status    |  Flags    |   Opcode     |        MBZ         |
+-----------------------------------------------------------+
|                       | <20-23>      |                    |
|         MBZ           | Control      |        MBZ         |
+-----------------------------------------------------------+
```

7.6.3.2 PLIRD Response -

The format of the response given to this command if the Flags RESP bit is on is specified below:

PLIRD Response Format

```
+-----------------------------------------------------------+
|                        Queue FLINK                        |
+-----------------------------------------------------------+
|                        Queue BLINK                        |
+-----------------------------------------------------------+
|                  Reserved for Software                    |
+-----------------------------------------------------------+
|  <0-7>    |  <8-15>   |   <16-23>    |                    |
| Status    |  Flags    |   Opcode     |        MBZ         |
+-----------------------------------------------------------+
|                       | <20-23>      |        | <28-35>   |
|         MBZ           | Control      |  MBZ   | PLI data  |
+-----------------------------------------------------------+
```

Opcode

The operation code for this packet is 7. This command does not generate an NI packet. Through this command, the port driver can read any quantity that the port processor itself can read over the PLI.

Control

These four bits specify the control function to be executed during the read cycle. The function executed is defined by the following table:

Value (octal)	Function
0	- Illegal
1	- RD.REG
2	- RD.REL.BUF
3	- RD.USED.LIST
4	- RD.XMIT.STATUS
5	- RD.REL.STATUS

If a command is selected which is not defined in this table as a legal function, a response will be generated which the status field set to "Illegal PLI function."

PLI data

This field (and the word that contains it) is present only in the response built for this command. This occurs when the RESP bit in the flags field is set. In the response, this field is the data that was read from the PLI by the execution of the RDPLI command.

It is not recommended for the driver to execute this command while other commands are pending or while packet reception is enabled. Careless use of this command can cause the normal port functionality to be compromised.

7.6.4 Read NI Station Address -

When executed, this command causes the NI station address to be read from the NI link module. In addition, some status mode bits are read. This data is returned in the response queue entry, if specified by the RESP bit in the flags of the command.

7.6.4.1 RDNSA Command -

The data for this command is returned in the response entry built if the RESP flag is set in the command's flags field when the command is executed. The format of this command is specified below:

RDNSA Command Format

```
+--------------------------------------------------------------+
|                        Queue FLINK                           |
+--------------------------------------------------------------+
|                        Queue BLINK                           |
+--------------------------------------------------------------+
|                   Reserved for Software                      |
+--------------------------------------------------------------+
|  <0-7>    |  <8-15>  |   <16-23>    |                        |
| Status    |  Flags   |   Opcode     |          MBZ           |
+--------------------------------------------------------------+
```

Opcode

The operation code for this packet is 8. If the response bit in the flags word is on, a response confirming the correct execution of the command will be built and placed onto the response queue anchored in the PCB

7.6.4.2 NSARD Response -

If built, the response to the RDNSA command is as follows:

RDNSA Response Format

```
+--------------------------------------------------------------+
|                        Queue FLINK                           |
+--------------------------------------------------------------+
|                        Queue BLINK                           |
+--------------------------------------------------------------+
|                   Reserved for Software                      |
+--------------------------------------------------------------+
|  <0-7>    |  <8-15>  |   <16-23>    |                        |
| Status    |  Flags   |   Opcode     |          MBZ           |
+--------------------------------------------------------------+
|<0---------------------------------------------31><32-35>|
|                 High Order NI Address             | MBZ  |
+--------------------------------------------------------------+
|                  Low Order NI Address             | MBZ  |
+--------------------------------------------------------------+
| <0---31>|   <32>   |    <33>   |     <34>     |    <35>     |
|   MBZ   |   ACRC   |    AMC    | H4000 Mode | PRMSC Mode |
+--------------------------------------------------------------+
|<0------------------15>|<16---------23>|<24-29>|<30-35>|
|           MBZ         | Ucode Version | #MCAT |  #PTT  |
+--------------------------------------------------------------+
```

High/Low NI Address

This value is the physical station address stored in the physical address RAM on the NI link board. It corresponds to the address of the NI link. The format for these words is identical to the NI station address format for the KL10 station address registers described in an earlier section. The initialization value of the physical address RAM is the value stored in the physical address ROM of the NIA.

ACRC

This bit indicates whether the NI link will discard incoming packets with CRC errors. If set the link will accept all incoming packets with CRC errors and place them on the unknown protocol type free queue if an entry is availible. If reset the link will discard all incoming packets with CRC errors.

The inialization value if this bit is zero

AMC

This bit indicates whether the NI link will accept All Multi-Cast packets, (if set), or will perform normal multi-cast address filtering (if reset.)

The initialization value of this bit is 0.

H4000 Mode

This bit shows the current state of the H4000 mode bit. This bit if set enables the heartbeat detection checking for the H4000 type NI bus tranceiver. The initialization value of this bit is 0.

PRMSC Mode

This bit if set indicates that the link is operating in promiscious mode. All packets detected on the wire will be interpreted as being addressed to this node. The initialization value of this bit is 0.

Ucode Version

This field gives the version of microcode loaded into the IPA20-L port. The initialization value of this field depends upon the microcode version loaded.

#MCAT

This field gives the number of Multi-Cast Address Table entries allowed.

#PTT

This field gives the number of Protocol Type Table entries allowed.

7.6.5 Write NI Station Address -

When executed, this command will set the physical address RAM on the NI link board, as well as set several mode bits for the datalink operation.

7.6.5.1 WRTNSA Command -

A response for this command is built only if the RESP bit in the command packet flags byte is on. The format of this command is illustrated below:

WRTNSA Command Format

```
+-------------------------------------------------------------+
|                        Queue FLINK                          |
+-------------------------------------------------------------+
|                        Queue BLINK                          |
+-------------------------------------------------------------+
|                   Reserved for Software                     |
+-------------------------------------------------------------+
|  <0-7>    |  <8-15>   |   <16-23>    |                      |
|  Status   |  Flags    |   Opcode     |         MBZ          |
+-------------------------------------------------------------+
|<0-----------------------------------------------31>|<32-35>|
|                 High Order NI Address               |  MBZ  |
+-------------------------------------------------------------+
|                  Low Order NI Address               |  MBZ  |
+-------------------------------------------------------------+
| <0---31>|   <32>   |    <33>    |    <34>     |    <35>     |
|   MBZ   |   ACRC   |    AMC     | H4000 Mode  | PRMSC Mode  |
+-------------------------------------------------------------+
|<0-------------------------------23> <24--------------35>|
|                 Must Be Zero          | # Retries Allowed |
+-------------------------------------------------------------+
```

Opcode

The opcode for this command is 9.

High/Low NI Address

This address is written to the physical address RAM on the NI link board. After the command completes, the NI link will accept packets with the physical address specified.

ACRC

This bit indicates whether the NI link will discard incoming packets with CRC errors. If set the link will accept all incoming packets with CRC errors and place them on the unknown protocol type free queue if an entry is availible. If reset the link will discard all incoming packets with CRC errors.

The inialization value if this bit is zero

AMC

Setting this bit will put the port into Receive All Multi-Cast mode. When enabled all multicast packets received are passed by the receive address filter.

The initialization value of this bit is 0.

H4000 Mode

If set by the driver, this bit will enable H4000 mode; this will enable heartbeat detection checking by the bus transceiver. This feature is to allow the NI link to accept tranceivers that do heartbeat checking (DEC only), or ones that do not.

PRMSC Mode

If set by the driver, this bit will enable promiscious mode. In this mode all packets seen on the NI cable will pass the received address filter. This is a diagnostic feature. Turning this mode on can cause significant degradation of system network performance, depending upon network load.

Retries Allowed

This field specifies how many retires of retrieable errors are to be attempted by the port before the error is declared uncorrectable. The default number of retries is 5.

7.6.5.2 NSAWRT Response -

The response format to the command listed above is given below:

NSAWRT Response Format

```
+--------------------------------------------------------------+
|                        Queue FLINK                            |
+--------------------------------------------------------------+
|                        Queue BLINK                            |
+--------------------------------------------------------------+
|                   Reserved for Software                       |
+--------------------------------------------------------------+
| <0-7>     | <8-15>    |   <16-23>    |                        |
| Status    | Flags     |   Opcode     |          MBZ           |
+--------------------------------------------------------------+
|<0-------------------------------------------31>|<32-35>|
|              High Order NI Address              |  MBZ  |
+--------------------------------------------------------------+
|               Low Order NI Address               |  MBZ  |
+--------------------------------------------------------------+
| <0---31>|   <32>   |    <33>    |    <34>     |     <35>      |
|   MBZ   |   ACRC   |    AMC     | E4000 Mode  | PRMSC Mode    |
+--------------------------------------------------------------+
|<0-----------------------------23>|<24--------------35>|
|            Must Be Zero           | # Retries Allowed  |
+--------------------------------------------------------------+
```

The bits for this response will be unchanged from the definitions for the WRTNSA command.

8.0 PORT REGISTERS

The Control/Status register is briefly defined in the following table. In the table, the symbol "*" indicates that the bit is not writable and is always read as zero. "R" means the bit is readable and "W" means that the bit is writable. Refer to the IPA20-L Port Hardware Specification for additional details.

BIT NO.	BIT DEFINITION	ACCESS KL10	ACCESS PORT
00	PORT PRESENT	R	*
01	DIAG RQST CSR	R	*
02	DIAG CSR CHNG	R	*
03	DIAG INITIALIZED	R	*
04	PI 00 L	R	R
05	RQST INTERRUPT	R	W
06	CRAM PARITY ERR	R	*
07	MBUS ERROR	R	*
08		*	*
09		*	*
10		*	*
11	IDLE LOOP	R	R/W
12	DISABLE COMPLETE	R	R/W
13	ENABLE COMPLETE	R	R/W
14		*	*
15		*	*
16		*	*
17		*	*

BIT NO.	BIT DEFINITION	ACCESS KL10	ACCESS PORT
18	CLEAR PORT	R/W	*
19	DIAG TEST EBUF	R/W	*
20	DIAG GEN EBUS PE	R/W	*
21	DIAG SEL LAR/SQR	R/W	*
22	DIAG SINGLE CYC	R/W	*
23		R/W	*
24	EBUS PARITY ERR	R/W	R
25	FREE QUEUE ERR	R/W	R/W
26	PORT ERROR	R/W	R/W
27	CMD QUEUE AVAIL	R/W	R/W
28	RSP QUEUE AVAIL	R/W	R/W
29	SPARE	R/W	R/W
30	DISABLE	R/W	R/W
31	ENABLE	R/W	R/W
32	MPROC RUN	R/W	R
33	PIA 00	R/W	R
34	PIA 01	R/W	R
35	PIA 02	R/W	R

Bit 26, PORT ERROR is used by the KLNI to inform the driver that an error has occured. When this bit is set by the port, the error logout area of the PCB has been written with valid error information. If port interrupts are enabled, an interrupt is submitted. Otherwise, the bits have the same definition as the KLIPA, the KL Computer Interconnect device microcode. This bit is cleared by reading the status register.

9.0 DATA FORMATTING MODES

NI20 supports 2 data formatting modes. These modes allow the operating system to specify how the data is packed in the host's memory on memory read operations and how the port should write the data into the host's memory on write operations. These modes are specified in each Buffer Segment Descriptor for that buffer segment. These modes are also specified for datagram transfers. The two modes are industry compatible and high density. In all of these modes, the most significant byte is left-justified so that normal PDP10-style byte pointers may be used to access the data. Note that it is not trivial to use standard PDP10-style byte pointers in high density mode. The port will always start reading or writing data with the most significant byte within the word. The number of bits within each byte follows the PDP-11 standard. The detailed descriptions of these packing modes follow.

9.1 Industry Compatible Mode

The following figure illustrates the industry compatible mode for mapping 8 bit NI bytes into 36 bit KL10 words.

```
                          INDUSTRY COMPATIBLE

                               KL10 WORD

  0          7  8          15 16         23 24         31 32   35
 +------------+------------+------------+------------+------+
 |            |            |            |            |      |
 |   Byte 0   |   Byte 1   |   Byte 2   |   Byte 3   | ZERO |
 |            |            |            |            |      |
 +------------+------------+------------+------------+------+
  7          0  7          0  7          0  7          0
                              PL1 BYTE
```

9.2 High Density Mode

The following figure illustrates the high density mode for mapping 8 bit NI bytes into 36 bit KL10 words. In this mode, it is important to realize that 9 bytes are packed into 2 36-bit words. Byte 4 is split across a word boundary; the most significant 4 bits are stored in bits 32-35 of the first word of the pair and the least significant 4 bits are stored in bits 0-3 of the second word of the pair. Because of the 4 bits stored in the most significant position of the second word, the remaining four bytes are stored right-justified in the second word. Because of this byte-packing mode, standard PDP10 byte pointers cannot be easily used to access all of the bytes in the buffer.

```
                HIGH DENSITY FORMAT

                       KL10 WORD PAIR
                     FIRST WORD OF PAIR
 0          7  8        15 16        23 24        31 32   35
+------------+------------+------------+------------+------+
|            |            |            |            | Byte |
|   Byte 0   |   Byte 1   |   Byte 2   |   Byte 3   |      |
|            |            |            |            |   4  |
+------------+------------+------------+------------+------+
 7         0   7        0   7        0   7        0  7    4
                         PLI BYTE


                    SECOND WORD OF PAIR
 0    3 4         11 12        19 20        27 28         35
+------+------------+------------+------------+------------+
| Byte |            |            |            |            |
|      |   Byte 5   |   Byte 6   |   Byte 7   |   Byte 8   |
|   4  |            |            |            |            |
+------+------------+------------+------------+------------+
 3   0  7         0  7         0  7         0  7          0
                         PLI BYTE
```

10.0 NI OPCODE SUMMARY

command/response	NI packet	dec	octal	binary
snddg/dgsnt	dg	1	1	0000 0001
ldmcat/mcatld	*	2	2	0000 0010
ldptt/pttld	*	3	3	0000 0011
rccnt/cntrc	*	4	4	0000 0100
/dgrcvr	dg	5	5	0000 0101
wrtpli/pliwrt	*	6	6	0000 0110
rdpli/plird	*	7	7	0000 0111
rdnsa/nsard	*	8	10	0000 1000
wrtnsa/nsawrt	*	9	11	0000 1001

11.0 PROGRAMMING NOTES

The following notes discuss various aspects of programming the LCG NI port.

11.1 Reset

When a port reset is issued to a running port, the hardware will be reset, and the port microcode PC will be set to zero so that the power-up, self-check initialization code will be executed. This guarantees that the port will enter a well defined state after the reset. It is important to realize that all of the port state will be lost; all state will revert to power-up state.

11.2 Initialization

When the LCG NI port is powered up, there is no valid microcode in the control RAMS (CRAMS); valid microcode must be loaded and started. This is the responsibility of the TOPS20 monitor or the NI20 diagnostic. During the initialization, the port microcode will perform a self-check test if it detects an error it will stop at a preset CRAM parity error. If there is no error, the microcode will also set the registers to initialization values, and enter the DISABLED state. When the port microcode sets the DISABLE COMPLETE bit in the CSR, the port driver should then write the PCE and the PI Level. It is not possible for the port to request a host interrupt until the PI level has been written with a valid assignment.

It is recommended that queue entries for the Unknown Protocol Type Free Queue be large enough to hold the largest legal Ethernet packet. Including flink, blink, and other overhead words, this is 388. words for an industry compatible mode received datagram.

*****FINIS*****