LCG CI Port Architecture Specification

Sean Keenan
Don Dossa
LSEG
MR1-2/E18
DTN 231-4463
January 1985
Revision 5.8

1.0 SCOPE

This document describes the architecture of the Computer Interconnect (CI) as it is implemented on LCG products. Included in this document is the interface the port provides to the operating system software.

2.0 REFERENCES

The following is a list of CI-related documents that the reader is assumed to be familiar with:

1. Computer interconnect Specification. Authors: D. Thompson, J. Buzinski, J. Hutchinson. This documents describes and specifies the implementation independent functional characteristics of the CI. Much of the material for the LCG-CI specification comes from this document.

2. VAX-11 CI Port Architecture Specification. Authors: W. Strecker, D. Thompson. This document describes the port/port driver interface for the CI port interfaced to the VAX family of processors.

3. CI20 Port Hardware Specification. Author: Elbert Bloom. This spec contains a full description of the hardware in the KL10 CI port option. Portions of that document have been reproduced here.

4. PILA Hardware Specification. Author: Shu-Shia Chow. This spec describes the register addresses and functions of the registers on the Packet Buffer and Link modules that can be accessed via the PLI.

5. KLCI ERROR SPEC. Author: Joe Holewa. This spec describes all the possible errors detected and generated by the port. It also defines algorithms for error retry and recovery.

3.0 GOALS

The LCG ports must provide a mechanism for LCG computers to interface to the CI bus. This requires that the LCG ports fully comply with the CI wire architecture specification.

4.0 NOTATIONAL CONVENTIONS

All bit numbers and word offsets in this document are in decimal. All field values within words are defined in octal. All bytes are always 8 bits long in this document with the most significant bit of any byte or field defined to be the left-most bit. It is also a goal to use the same terminology and definitions as the VAX CI port architecture spec to avoid confusion.

5.0 INTRODUCTION

The LCG CI port architecture is the operating system interface to a Computer Interconnect (CI). The CI is a multi-drop bus used to closely link computer systems and intelligent mass storage controllers. Characteristics of the CI include:

1. High bandwith of 70 Megabits/second,

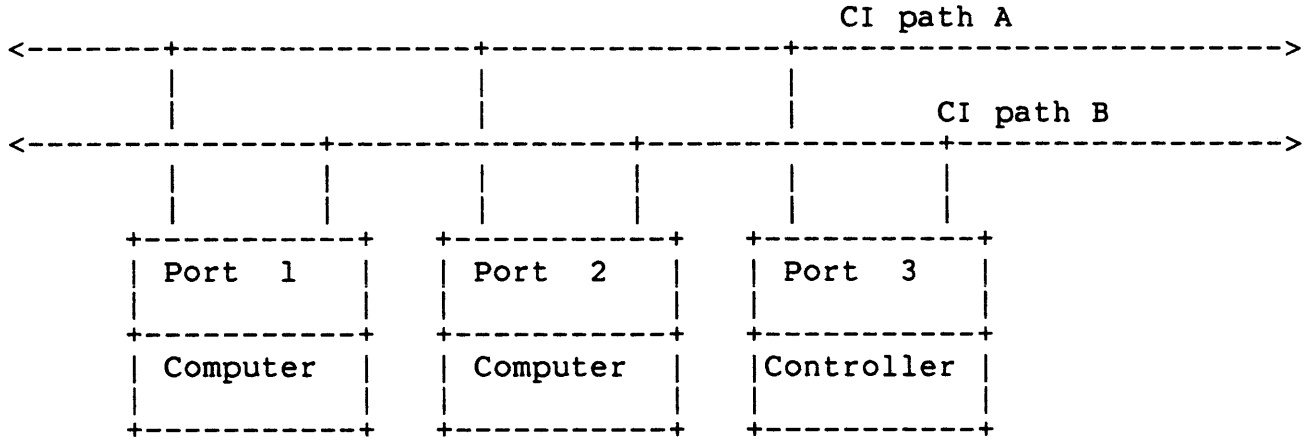
2. Packet oriented transmissions,

3. Low error rates,

4. Immediate acknowledgement of successfully-received packets.

Up to 16 ports may be connected onto a single CI bus.

The following figure shows a number of computers and intelligent mass storage controllers connected to a CI. A single CI consists of 2 paths. A single system may be connected to more than one CI.



6.0 ARCHITECTURE OVERVIEW

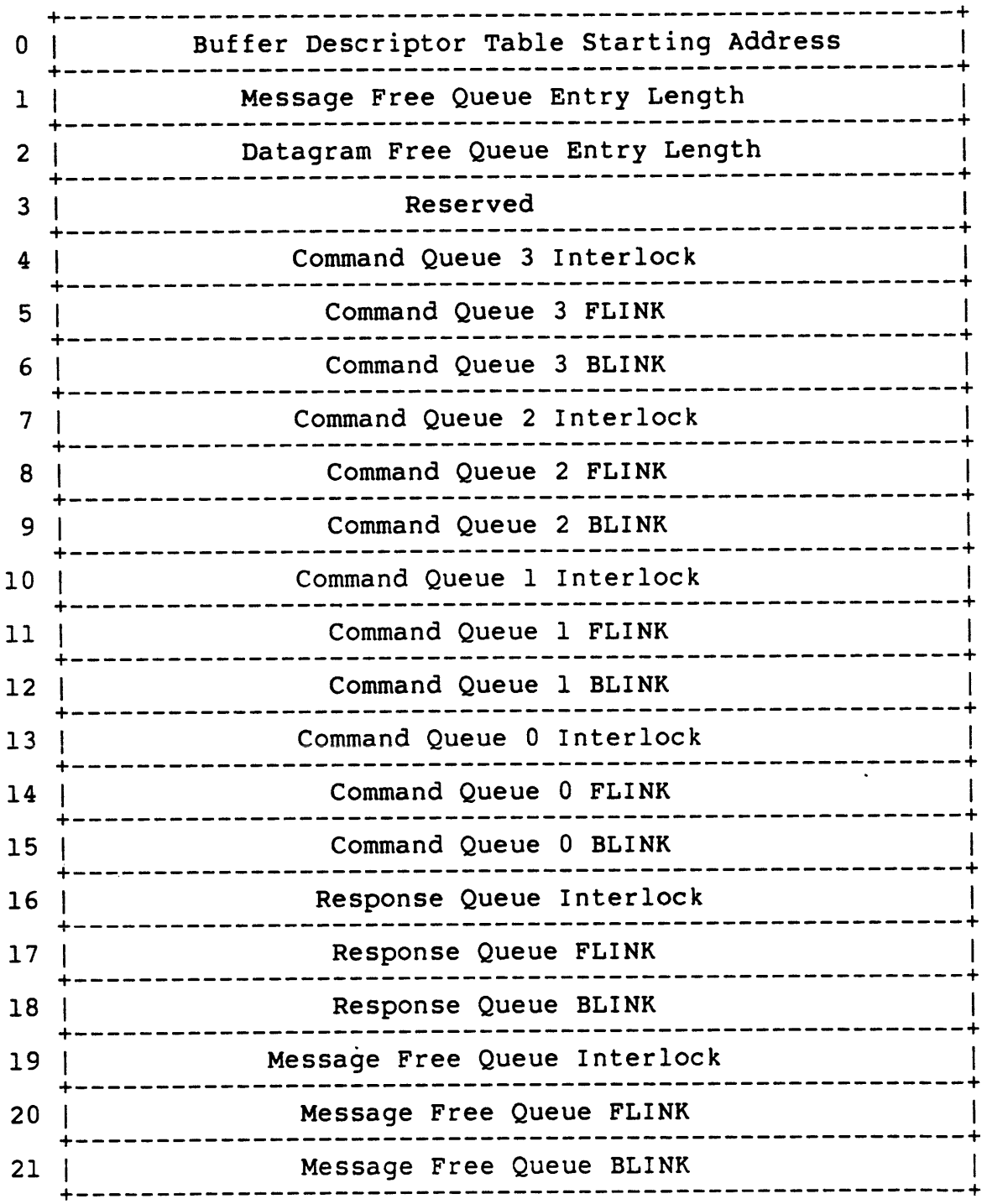
The port architecture implemented on LCG computers will comply with the corporate CI architecture specification electrically at the physical link level. It will also comply at the data link level in terms of packet formats, acknowledgements, path selection, sequentiality of data packets, error reporting and recovery algorithms, datagram service, and Virtual Circuit service.

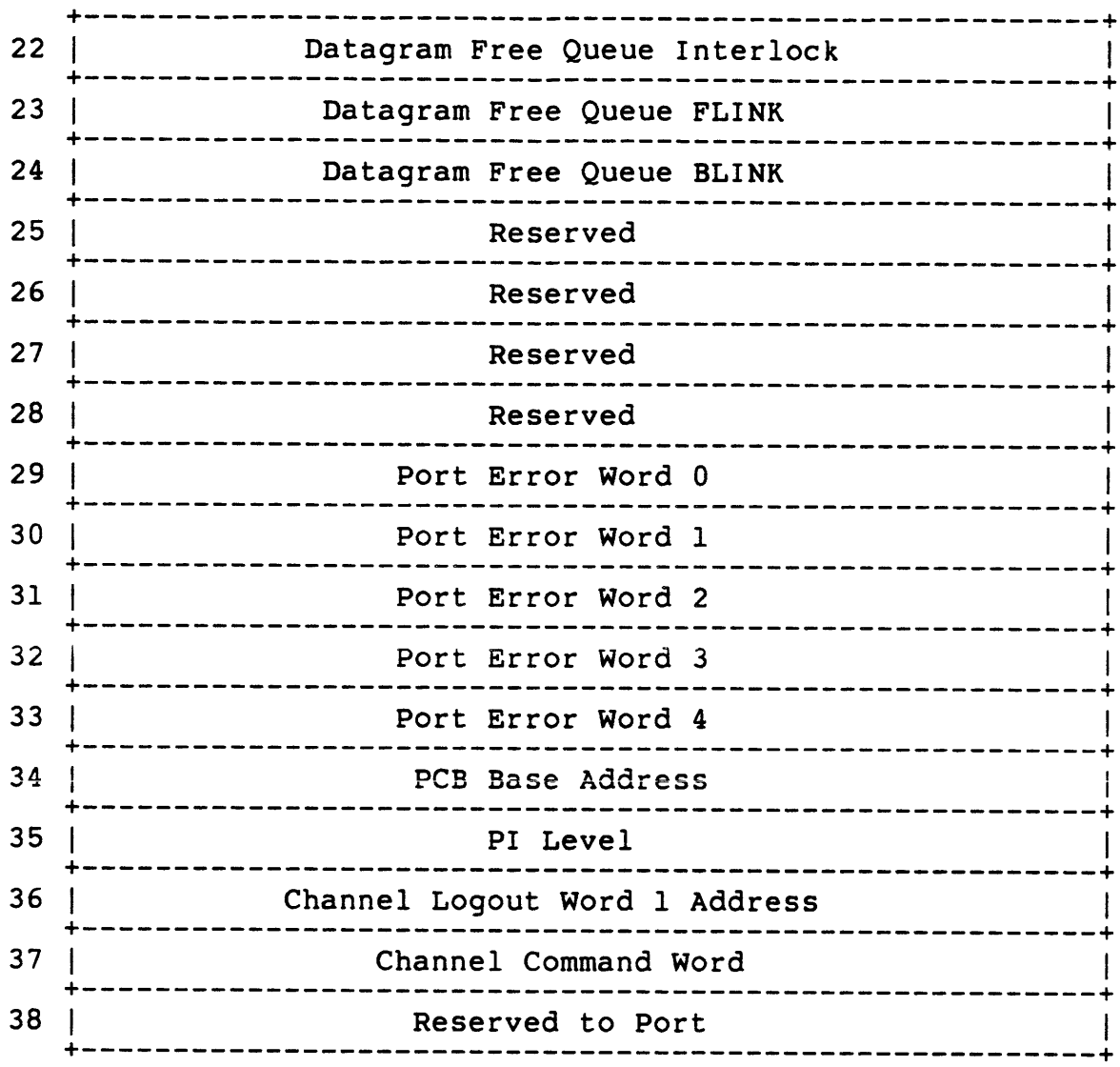
The mechanism for the host to communicate with the port is by use of the queued protocol. The basic data structure of the queued protocol relies on 7 different doubly-linked queues. The queues are referred to as Command Queues 0-3, the Response Queue, and two free queues, the Datagram Free Queue and the Message Free Queue. When the host wishes to send the port a command, it places an entry on the tail of one of the four command queues. The port communicates with the host by linking entries onto the tail of the response queue. The host processes the response queue entry by delinking the queue entry from the head of the response queue. Both the host and port use the free queues as a source of available queue entries and as a repository of processed and discarded queue entries. The data structure that ties the queue link words together is the Port Control Block (PCB).

6.1 Port Control Block (PCB)

The mechanism where the host and the port share the queue structures is controlled by the Port Control Block. The Port Control Block is a data structure that exists in the physical memory space of the host computer. Both the host and the port read and write the data in the PCB. The PCB contains the link words for the queues and other control information. There is exactly one unique PCB for each CI port; the ports may not share PCB's. The port driver has the responsibility of initializing all entries in the PCB. All reserved entries must be zero and all of the queue FLINK and BLINK words must be set to valid values. The format of the PCB follows:

PORT CONTROL BLOCK (PCB)





The Buffer Descriptor Table (BDT) base address contains the physical address of the first word of the buffer descriptors. Buffer descriptors contain all of the information necessary to specify to a port where in the host system's memory a data buffer is located and what access methods are to be used. Buffer descriptors are described in detail in a separate section of this document.

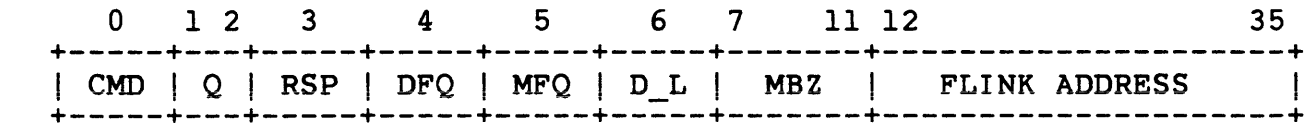
The CI20 requires special KL10 microcode support to allow the CI20 to perform a memory increment operation using read-pause-write memory references. This is needed to allow the port to interlock the queues.

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.

Error Words 0,1 (words 29,30) 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.

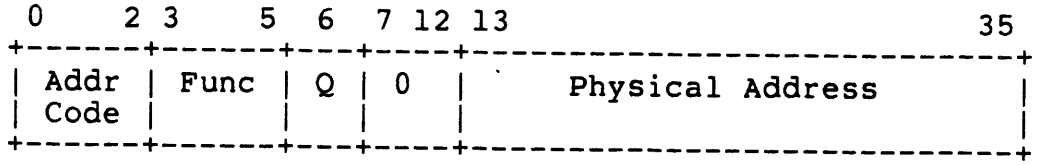
The format of Error Word 0 is:



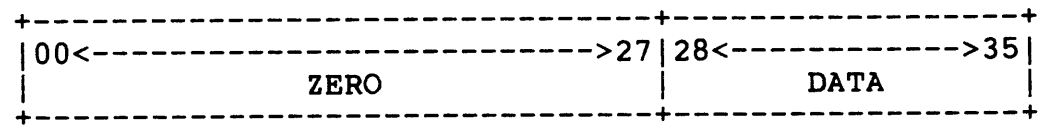
BITS	NAME	DESCRIPTION
0	CMD	Error occurred while touching a command queue entry. The queue with the error is in QUEUE.

1-2	QUEUE	This is the command queue that had the error. These bits are only valid if the CMD bit is on. 00 = CMD QUEUE 0 01 = CMD QUEUE 1 10 = CMD QUEUE 2 11 = CMD QUEUE 3
3	RSP	This bit is on if the error occurred while the port was attempting to build a response queue entry.
4	DFQ	This bit is on if the error occurred while the port was touching a command on the datagram free queue.
5	MFQ	This bit is on if the error occurred while the port was touching a command on the message free queue.
6	D_L	This bit is on if the error occurred while the port was linking a command to a queue. This bit is off if the error occured while the port was delinking a command from a queue. This bit is valid only with bits 0,4 and 5.
7-11	MBZ	These bits will be zero.
12-35	FLINK ADR	This is the address of the FLINK word of the queue entry in question.

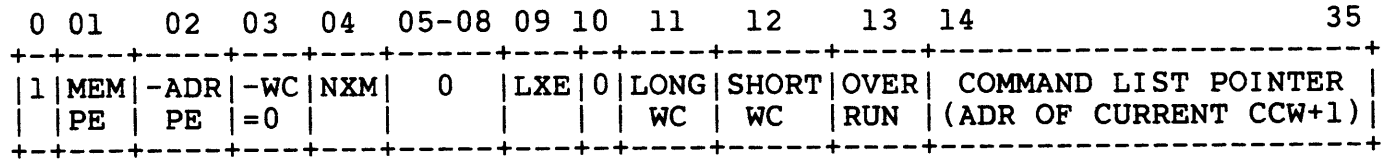
Error Word 1 (word 30) contains the API function word that the port processor used to access memory when the memory error occurred. This word is written here in the same format as it appeared on the EBUS. The format of this word is:



Error Word 2 (word 31) contains the register data on Transmitter or Receiver spurious attention. The format of Error Word 2 is:

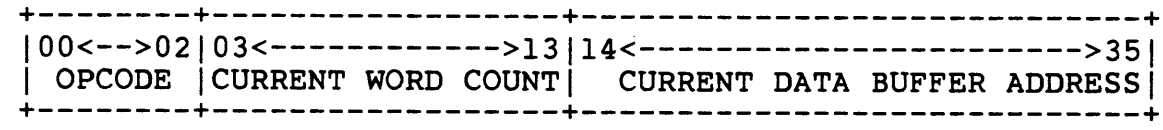


Error Word 3 (word 32) contains the Channel Logout Word 1 written by port on any kind of channel error detected during or immediatly after, a DMA transfer. The format of Error Word 3 is:



BITS	NAME	DESCRIPTION
01	MEM PE	Memory Parity Error.
02	-ADR PE	Not Address Parity Error.
03	-WC=0	Chan Word Count did not = 0 when chan did a store to EPT.
04	NXM	Chan ref non exist mem.
09	LXE	Error detected after port term transfer, Chan aborts next transfer.
11	LONG WC	Port comp Xfer, But word count in CCW not reached.
12	SHORT WC	Chan Xferred data spec by CCW, But port still has data.
13	OVER RUN	If dev read, Port sent data but chan buff were full. If dev write, Port req data but chan buff were empty.

Error Word 4 (word 33) contains Channel Logout Word 2 written by port on any kind of channel error detected during or immediate after, DMA transfer. The format of Error Word 4 is:



Word 34 of the PCB is the address of the first word of the PCB; the CI20 has no other way of finding the PCB.

Word 35 contains the PI Level that the CI20 is assigned.

Word 36 contains the address of the Channel Logout Word 1. The Channel Logout Word is used in CBUS Error recovery processing.

Word 37 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 in the appropriate EPT location corresponding to the RH20 backplane slot where the CI20 is installed.

Word 38 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 CI20 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 (words 34-36 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 the current 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 States And Programming

6.2.1 Port States -

There are 3 defined states in which the port can be. They are:

1. Uninitialized - The port is not running. This is the power up state. The port enters this state after a power-on or a master reset. The port may exit this state only if valid microcode is loaded into the port and the port clocks are started.

2. Disabled - The port is running, but it is not accepting CI packets. The port will process entries from the command queues with opcodes of 200 or greater. This state can only be entered from the uninitialized state by starting the port microcode running and setting the Disable bit (Bit 30) in the CSR. The microcode will put the port into this state and signal it by setting Disable Complete (Bit 12) in the CSR. The Disabled state is entered from the Enabled state through a command from the host to enter this state or when the port microcode detects a non-recoverable internal port error.

3. Enabled - The port is fully functional; it is processing commands and CI packets. This is the normal state for the port. This state can only be entered from the Disabled state via command from the host (Bit 31 in CSR).

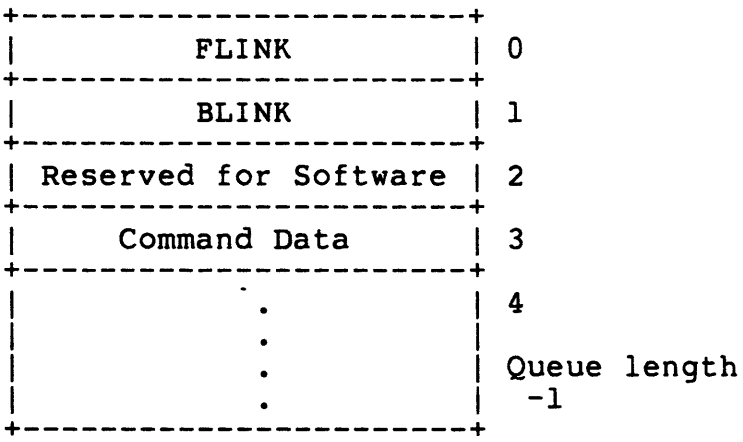
6.3 Queue Structures

There are 7 queues used by the host and port. The four command queues contain commands for the port that it has not processed yet; the port will delink and execute queue entries from the head of these queues. The priority ordering of the command queues is from Command Queue 3 to Command Queue 0. The port will examine Command Queue 3 and process all commands on this queue before moving to a command queue of lower priority. When Command Queue 3 is empty, the port will examine Command Queue 2 for any valid commands and process the first command it finds. From this point on, the port will process a command from a lower priority queue only after it has determined that no command exists on a higher priority queue. The port determines that a command has been placed upon a previously empty command queue by examining a bit in a control register that is set by the host when it has placed a

command upon a previously empty command queue. This guarantees that any command placed on a command queue of higher priority will be processed before the next command on a lower priority queue.

Similarly, if a response queue entry is required, the port will place responses onto the tail of the response queue when it has finished processing either a locally generated or remotely generated command. The host will process response queue entries by delinking them from the head of the response queue, processing the results, and placing the discarded entries on the tail of one of the free queues. If a response does not have to be generated by the port, it will place the discarded entry onto the tail of the free queue. All datagram-class commands and responses use the Datagram Free Queue as both a source and sink for queue entries. All commands and responses that are executed under Virtual Circuit control use the Message Free Queue. The description of Virtual Circuit commands is covered in a later section.

The general format of a queue entry is:



The length of a queue entry is specified by the host operating system to the port via words in the Port Control Block. Because the queue FLINK and BLINK words are interlocked, it is required that all queue insertions and deletions follow the proper protocol to avoid race conditions between the port driver and the port microcode. The reserved for software word is never written by the port.

6.4 Port Commands And Responses

6.4.1 Command Processing -

When the port driver places a command upon one of the command queues, the port microcode will delink the first command in a particular 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.

The PORT field in a command refers to the destination port; this field will be ignored if the command does not reference a remote port.

6.4.2 Self Directed Commands -

Commands that have the destination port field equal to the port's own number will cause the port to perform all of the normal functions including Virtual Circuit checking, if required by the packet type. The port will use internal loopback mode on self directed commands. The port will receive its own message and process it normally. On LOOPBACK commands, it is necessary for the port driver to append the 32-bit CRC characters at the end of the data transfer. In this case, the 4 bytes of CRC must directly follow the last byte of text, but the 4 CRC characters are not included in the text byte count specified in datagrams. It must be recognized that while in internal loopback the port is unable to receive packets from the CI wire.

6.4.3 Responses -

All locally-generated commands that require a response will cause the generation of a response packet that will be linked onto the tail of the response queue. Remotely generated packets will also cause responses to be generated. This is true for all datagrams and messages, configuration packets, maintenance packets, unknown or illegal type packets, and data transmission confirmation packets.

The port allows the port driver to determine which responses are generated because of the execution of locally-generated commands and which response queue entries are generated because of

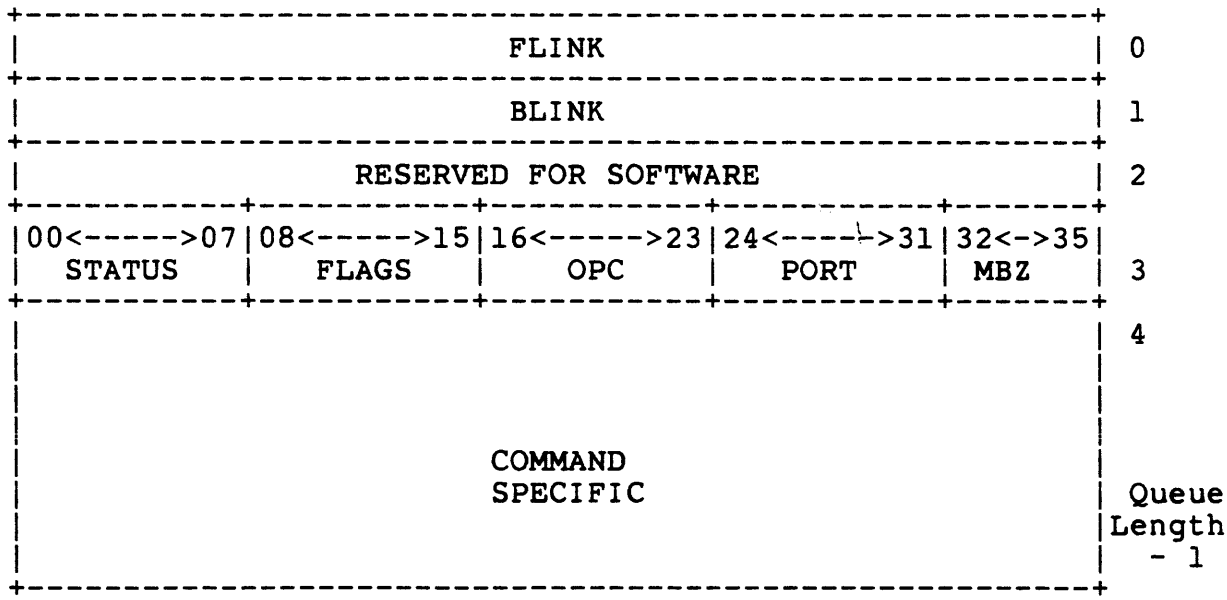
reception of packets over the CI wire. All commands which the port driver builds and the port executes are called locally-generated commands; the corresponding response queue entries have the identical opcode and are referred to as locally-generated responses. Response queue entries which are generated because the port received a packet over the CI are referred to as remotely-generated responses. The port will always add a decimal 32 to the value of the opcode field in remotely-generated commands when building the response queue entry.

In locally-generated responses, the PORT field is not modified by the port. In remotely-generated responses, the PORT field contains the port number from which the packet was received.

7.0 PACKET TRANSMISSION CHARACTERISTICS

7.1 Basic Command Queue Format

The basic queue entry format is the same for all command queue entries. The format of this queue entry follows:



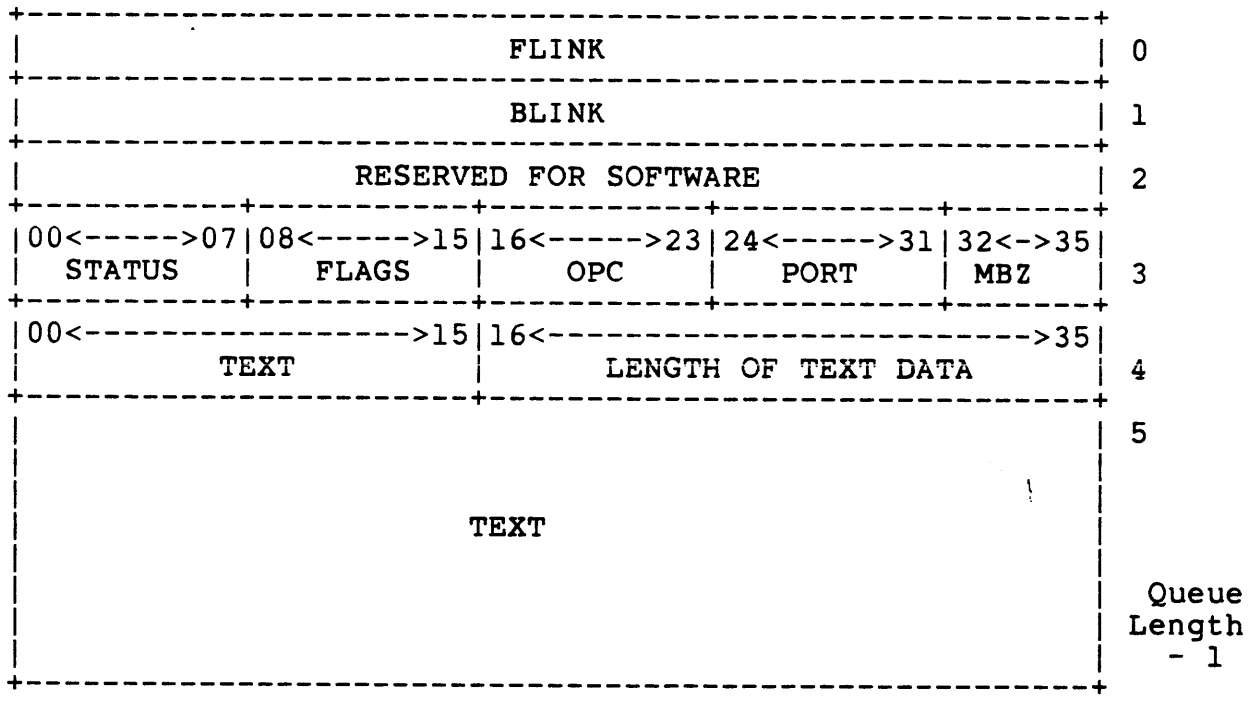
WORD:BITS	NAME	DESCRIPTION
3:0-7	STATUS	These bits must be zero when the port driver places this command upon the port's command queue. The port microcode will report the status of this command via this field when the port moves this command to the corresponding response queue.
3:8-15	FLAGS	This field is a collection of bits that allow command modifiers to be given to the port. These bits are defined below for all commands.
3:8	PACKET SIZE	This bit defines the base size of the data packets. If the bit is off, the base size is 512 bytes; if the bit is on, the base size is 576 bytes. This bit definition is only valid for data packets. Refer to the definition below if the packet is a datagram or message packet.
3:8	FORMAT	If this bit is off, LCG ports assume the datagram and message data is packed in industry compatible mode. If the bit is on, the data is in high density mode. This bit definition is only valid for datagram and message packets; refer to the above definition for data packets.
3:9-11	M	These bits determine the actual size of the data packets to be sent. The packet size used = (PACKET SIZE) * (M + 1). These bits are valid only for data packets.
3:13-14	PATH SELECT	These are the Path Select bits. PS = 0 => Automatic path selection PS = 1 => Select path A PS = 2 => Select path B PS = 3 => Invalid

3:15	RESPONSE (R bit)	If this bit is on, the port will always generate a response packet for this command. If the bit is off, the port will generate a response only if there was some error encountered during the processing of the command.
3:16-23	OPCODE	This is the opcode field for the command. Refer to the individual commands for details.
3:24-31	PORT	This is the source or destination port number.
3:32-35	MBZ	The port expects these bits to always be zero.

7.2 Datagram Service

The datagram service supplied by the port microcode provides a best-effort delivery service of messages from one port to another port over the CI. The text portion is transmitted over the CI from left to right within each 36 bit word. The first 2 bytes of text are stored in the same word as the text length, which is not transmitted over the CI. These first 2 text bytes are also transmitted from left to right. All datagrams received over the CI are also written to host memory from left to right in the order they are received.

The format of the send datagram command is:



WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 1 octal (SNDDG) 41 octal (DGREC)

4:00-15	TEXT	The left-most byte in this word is the first text byte to be sent/received over the CI wire. The number of bytes is determined from the length field and the packing format is determined from the FLAGS field.
4:16-35	LENGTH	This 20-bit quantity is the number of bytes in the message. This number must be between 0 and the lesser of the remaining number of bytes in the command queue entry and 4089. Note that the port hardware does not currently support packets of greater than 1017. bytes. This field is not sent over the CI.
5:0-??	TEXT	The remaining text.

At the completion of the send datagram command, the port microcode will determine if it should build a response queue entry for this command.

The response is called a Datagram Sent Response (DGSNT). If the Response bit (R bit) is set in the FLAGS words, the port will always build a response entry. If the R bit is off for this particular command, no response will be built unless an error occurred during the processing or transmission of the packet. Any error condition always causes the port to build a response packet. The format of the response packet is similar to the send datagram packet; the only difference is that the STATUS field is returned with a non-zero value to indicate the type of failure. Refer to the section on responses for a detailed description of all of the allowable values for the STATUS field.

If no response is to be built, the SNDDG command will be linked onto the tail of the datagram free queue.

A remotely-generated Datagram is a DGREC.

7.3 Virtual Circuits

A Virtual Circuit is the mechanism that is used to provide a higher quality delivery service for messages than the datagram service. Delivery of packets under Virtual Circuit control is guaranteed to be sequential, error-free, and non-duplicated. The circuit is constructed of 3 state variables in the sending and receiving nodes.

Virtual Circuits are maintained on a per-node basis. Each circuit is independently controlled. Each pair of nodes has a Virtual Circuit state with respect to each other. The state of the circuit is determined by three bits internal to the port as specified by the Set Circuit (SETCKT) Command. These bits are the circuit state (CST), the sending sequence number (NS) and the receiving sequence number (NR). The CST bit is set to a one for each port when the Virtual Circuit is opened between them. NS and NR are used to guarantee delivery, sequentiality and non-duplication. The NS is the number of the next packet to be sent (or the present packet being transmitted). The NR is the number of the next packet to be received.

When a packet is to be sent under Virtual Circuit control, the transmitting port first checks to make sure that the Virtual Circuit is opened between the two ports. If it is not, the packet will not be sent over the CI wire, but an appropriate error response will be generated by the port. If the Virtual Circuit is opened, the transmitting port loads the value of NS into the defined bit in the FLAGS field. When successful transmission is determined NS is complemented.

In the receiving node, the state of the Virtual Circuit is determined by the CST bit, and if the Virtual Circuit is opened between the two ports, then the value of NS carried in the packet is compared with the state of the NR bit for the circuit. If they are equal, the packet is accepted and NR is complemented. If the values are not equal, the packet is discarded.

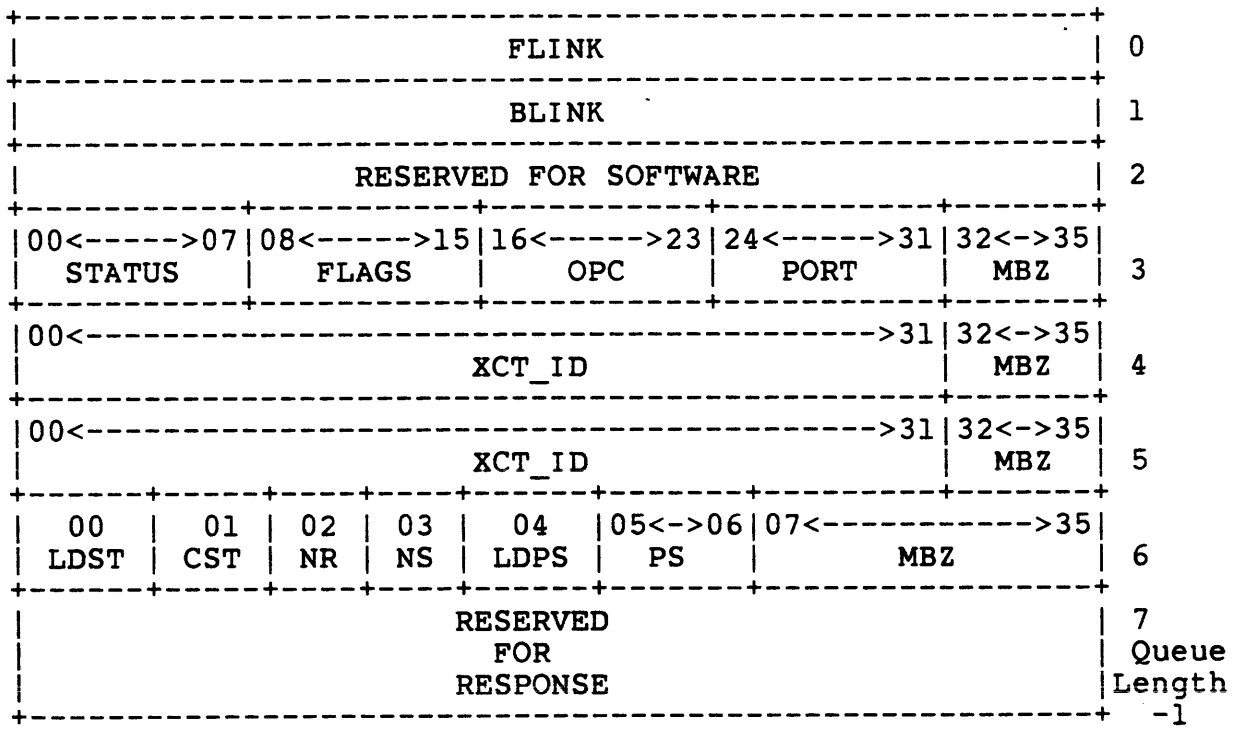
The Virtual Circuit will be closed if any transmission fails; any condition that may result in the disruption of sequentiality also causes the port to close the Virtual Circuit. In addition, a port may close a virtual circuit at any time.

The port microcode also maintains the status of the CI paths. This information is kept with each Virtual Circuit. The port driver determines the path availability when it establishes a Virtual Circuit.

7.4 Path Selection

Because there are normally two paths available on the CI ports, there is a mechanism to determine on which path a CI packet should be sent. There are two modes - Select and Automatic. Select mode is used when the port driver wants a particular path to be used. Automatic mode is used when the port driver does not care which path is used. The port microcode will implement a mechanism in which the port will randomly select a path in the automatic mode.

The Virtual Circuit state between any two ports is set via the Set Virtual Circuit (SETCKT) command. The format of the command is:

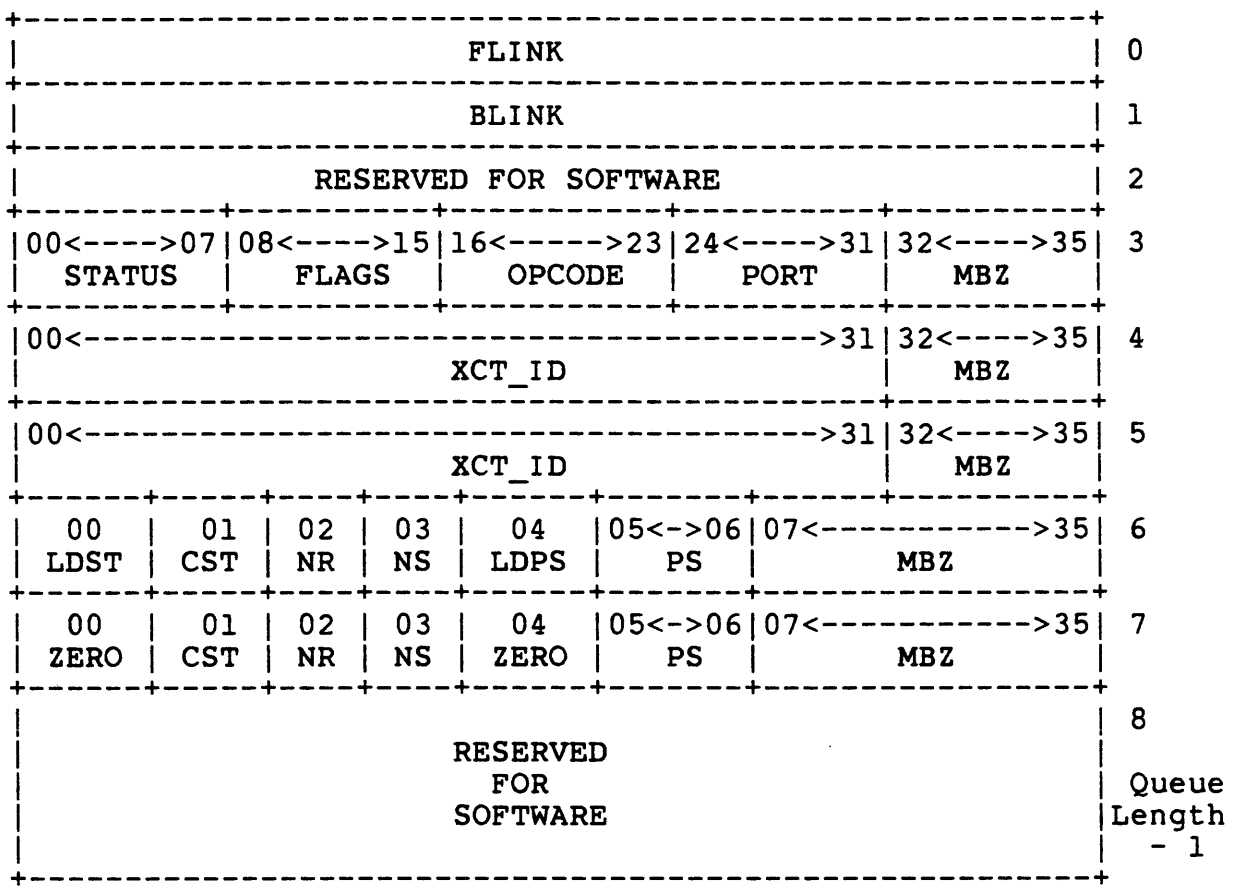


WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 200 octal (SETCKT).

4-5:0-31	XCT_ID	The XCT_ID is a 64-bit, 2 word quantity that is reserved for use by the port driver. The port will never modify this quantity but preserves it so that response queue entries can be associated with a particular command.
6:00	LDST	If this bit is set, the CST, NR and NS bits are set according to bits 1-3.
6:01	CST	Virtual Circuit State CST = 0 => Circuit closed. CST = 1 => Circuit opened.
6:02	NR	Receiving Sequence Number
6:03	NS	Sending Sequence Number
6:04	LDPS	If this bit is set, the Path Select bits are set according to bits 5-6.
6:5-6	PS	Path Select Bits PS = 0 => Neither path allowed.. PS = 1 => Path A allowed. PS = 2 => Path B allowed. PS = 3 => Both paths allowed.
6:7-35	MBZ	These bits must always be zero.

If the R bit is set in the FLAGS field, then the port will generate a response by moving the SETCKT command queue entry to the response queue. The port microcode will also return the current state of the specified Virtual Circuit in word 7 of the packet.

The actual format of the CKTSET command is:



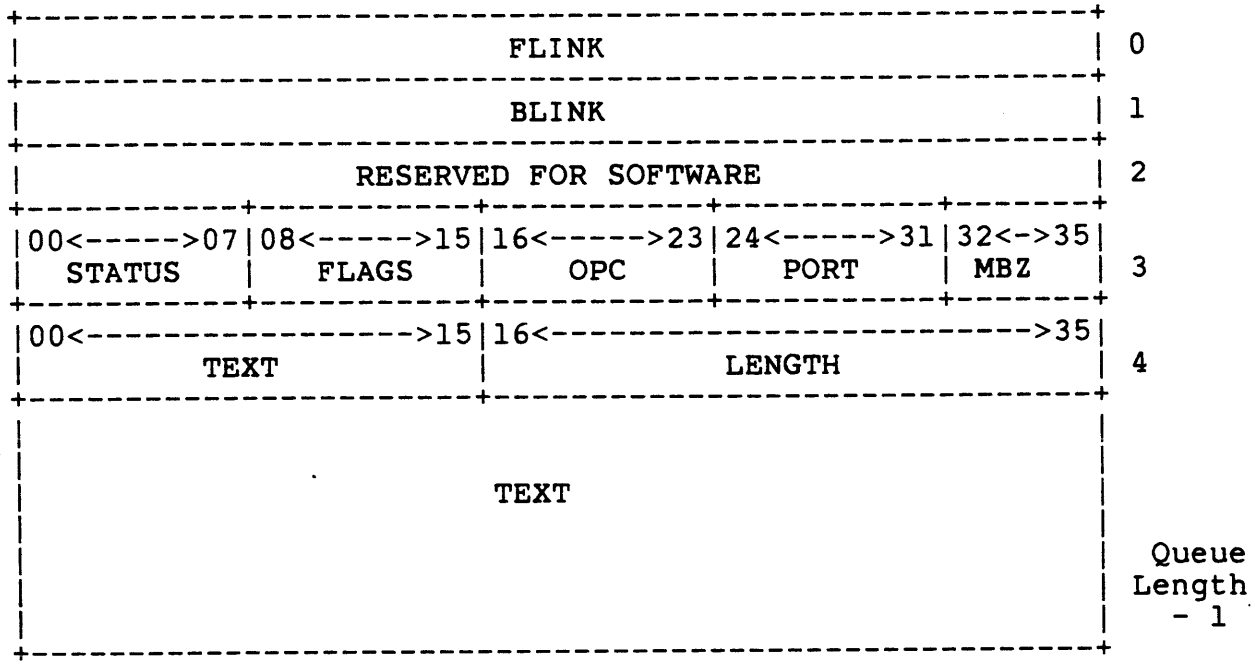
WORD:BITS	NAME	DESCRIPTION
7:00	ZERO	This bit will be zero.
7:01	CST	The current state of the Virtual Circuit between this port and the port specified in the PORT field is returned here.
7:02	NR	This is the current state of the NR bit.

7:03	NS	This is the current state of the NS bit.
7:04	ZERO	This bit will be zero.
7:5-6	PS	This is the current state of the paths.
7:7-35	MBZ	These bits will be zero.

8.0 MESSAGE TRANSMISSION

The message mechanism provides highly reliable delivery of single message packets under Virtual Circuit control. Messages are of variable length, ranging from 0 to 4089 bytes in textual length. The maximum size messages that may be sent from one port to another is determined at a higher level protocol. As with datagrams, the bytes will be transmitted over the CI from left to right within each host memory word. The same left to right order will also apply to all received messages.

To send a message, the port driver places a command upon one of the command queues in the host memory. The format of the basic message command (SNDMSG) is:



WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 2 octal (SNDMSG)(MSGSNT) 42 octal (MSGREC)

4:00-15	TEXT	The left-most byte in this word is the first text byte to be sent over the CI wire. The number of bytes is determined from the length field and the packing format of either industry compatible or high density is determined from the FLAGS field.
4:16-35	LENGTH	This 20-bit quantity is the number of bytes in the message. This number must be between 0 and the lesser of the remaining number of bytes in the command queue entry and 4089. Note that the port hardware does not currently support packets of greater than 1017. bytes. This field is not sent over the CI.
5:0-??	TEXT	The remaining text.

The format of the Message Sent (MSGSNT) response is the same as the SNDMSG command except that the STATUS byte will contain the result of the command execution. The description of the STATUS byte contains all of the allowed values for this field.

A remotely-generated response is a MSGREC.

9.0 DATA TRANSMISSION

Data packets provide a mechanism to allow the CI ports to move large amounts of data with a minimal amount of port driver overhead. Data transfers use the Virtual Circuit mechanism to provide a high quality transmission medium. All data transfers reference named buffers in hosts' memories. All commands contain the buffer names of the sending and receiving buffers and offsets which the ports use to determine where data is read from and written to. The buffer names and offsets must be agreed upon between the hosts' port drivers at a higher level of protocol. LCG hosts must also set up the buffer descriptor chains prior to transferring data. The Virtual Circuit must also be opened before the data transfer command is executed by the port.

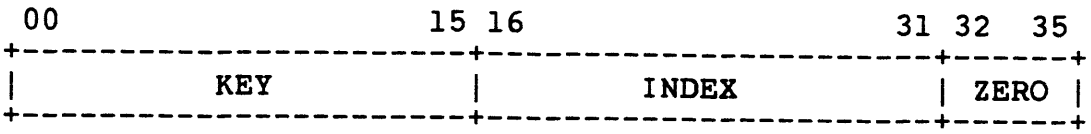
9.1 Buffer Descriptors

Named memory buffers in the physical address space of the host computers are described by buffer descriptors. Each buffer descriptor consists of a header, which uniquely defines a named memory buffer. Associated with each buffer header descriptor is a linked chain of buffer segment descriptors, which uniquely define contiguous segments of the buffer. The buffer header descriptor and all associated linked buffer segment descriptors constitute a buffer descriptor.

The buffer descriptors are assembled and inserted into a Buffer Descriptor Table (BDT) by the port driver. The Buffer Descriptor Table also resides in the host computer's physical address space. Since the Buffer Descriptor Table is accessed via the buffer names, it is not necessary that the entire table be physically contiguous; the only restriction on the placement of Buffer Header Descriptors and Buffer Segment Descriptors is that they must be allocated on 4 word boundaries.

A buffer name is a 32-bit quantity which is passed between ports via the name fields of the commands and their associated control packets or data packets. The name fields are supplied and inserted into the applicable commands by the port drivers. The port uses the name fields to access the buffer descriptors for commands which are associated with data transfers between named buffers.

The format of a buffer name, as it appears in the host computer's memory, follows:

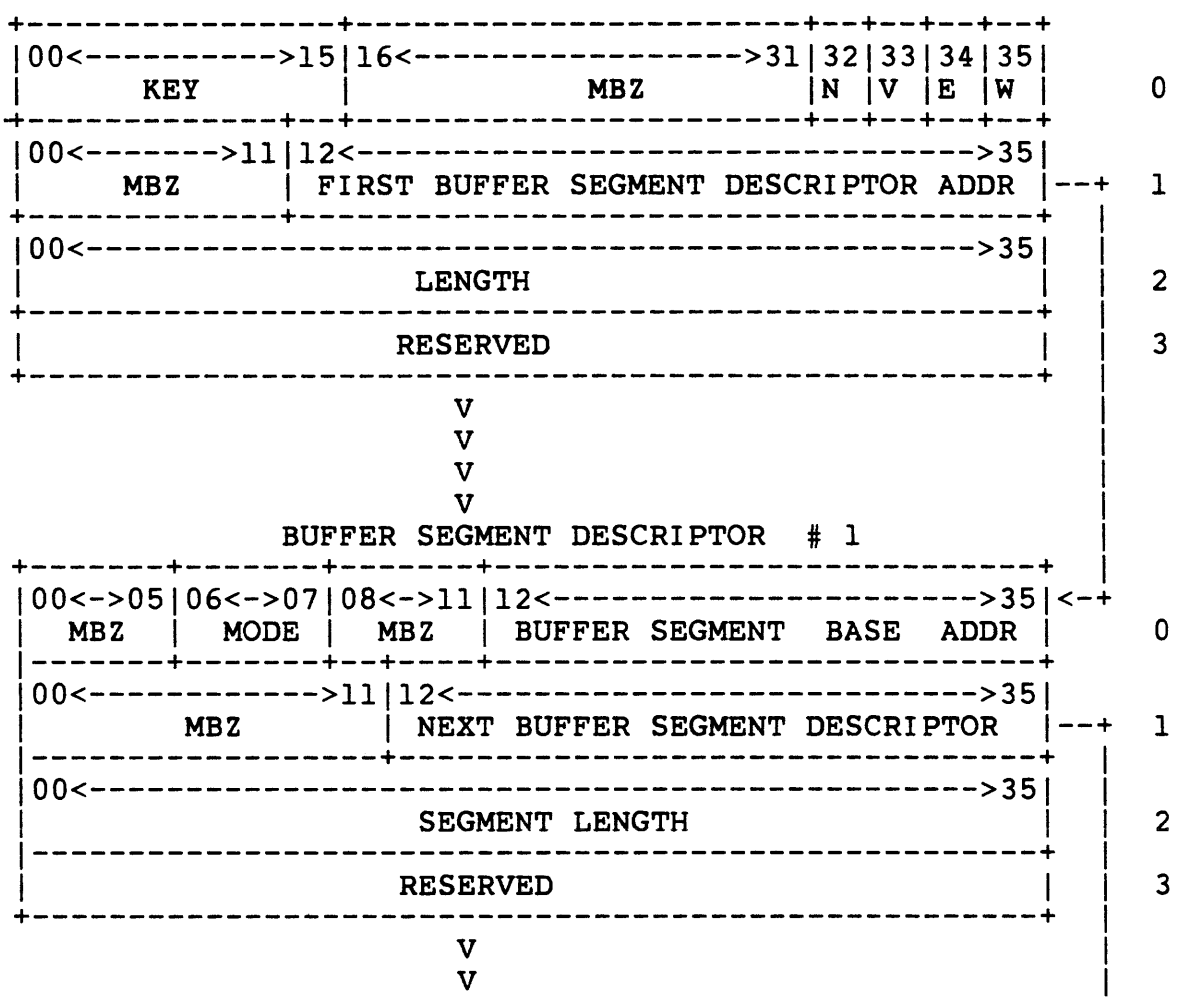


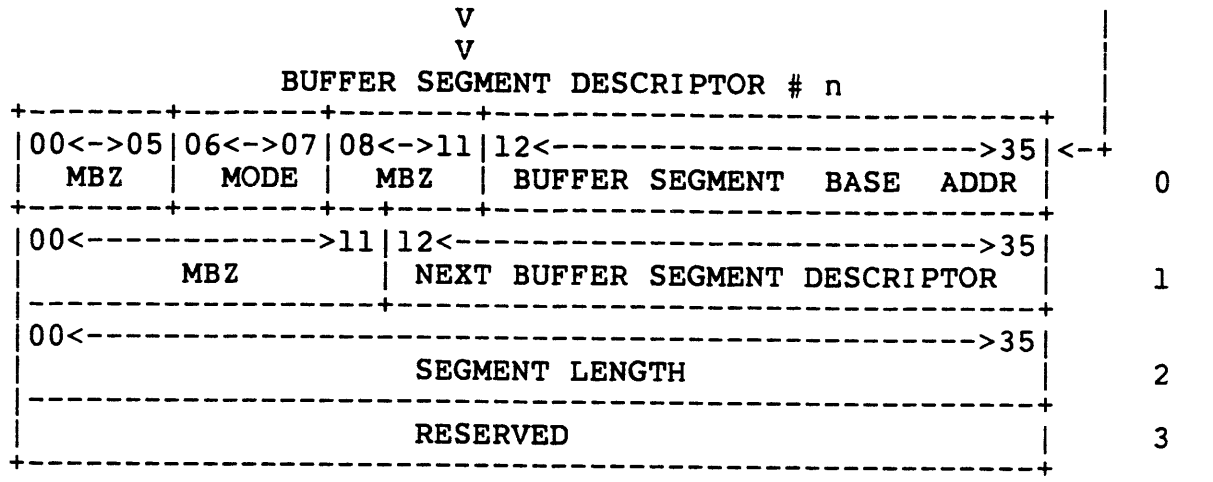
The INDEX field is used by the port as a word offset from the start of the buffer descriptor table to find the buffer header descriptor for this buffer name. The port will add the contents of the index field to the contents of word 0 of the PCB. The port microcode will then compare the KEY field fetched from the BHD with the KEY field supplied with this buffer name. This is a check on the value of the INDEX field. If the keys do not match, then the buffer name received from either a command queue or a packet from the CI bus is in error and appropriate error processing is initiated. If the keys do match, the port microcode will continue processing, knowing that this is the correct BHD to associate with the supplied buffer name.

Buffer Header Descriptors contain all of the information that the port microcode needs to find the buffers that contain the data to transfer over the CI. The BHD contains the start of the buffer in the host computer's physical address space, its length, some access control bits, and pointers to buffer segment descriptors (BSD) which contain further information. Each buffer name has no more that 1 BHD associated with it and the BHD points to the first BSD which in turn points to the first physically contiguous segment of the data buffer. Each succeeding BSD points to another physically contiguous segment of data. There are as many BSDs as necessary to completely describe the data buffer. The VALID bit in the BHD provides a mechanism for the port driver to inform the port that this BHD and its BSDs are valid and opened. The port will always assume that each BHD and all BSDs start on 4-word boundaries; this allows the port to access the data in the most efficient manner possible.

The actual format of a BHD is:

BUFFER HEADER DESCRIPTOR





BUFFER HEADER DESCRIPTOR FIELD DEFINITIONS

WORD:BITS	NAME	DESCRIPTION
0:0-15	KEY	This is the KEY that the port microcode compares with the KEY supplied in the buffer name. These keys must match for the port to continue processing the request for this buffer name.
0:16-31	MBZ	These bits must be zero.
0:32	NO CLEAR VALID	The port driver software must set this bit if it does not want the port microcode to clear the VALID bit.
0:33	VALID	The port driver software must set this bit to declare that this BHD is opened. Once this bit is set, the port microcode will assume that no further changes to this BHD or its associated BSDs will be made by the port driver software. This allows the port microcode to internally cache this data. The VALID bit will be cleared by the port microcode when the DMA transfer completes without error.

0:34	ERROR	This bit must be initially cleared by the port driver software when it marks the buffer as open by setting the VALID bit. The port microcode will set this bit to indicate to the port driver that an access violation or buffer length overflow occurred using this BHD or one of its BSDs.
0:35	WRITABLE	This bit is set by the port driver when the BHD is opened if the port microcode is allowed to write data into this buffer.
1:0-11	MBZ	These bits must be zero.
1:12-35	BSD ADDRESS	This is the address of the first word of the first buffer segment descriptor associated with this BHD. This address is a physical address within the host computer's memory space.
2:0-35	LENGTH	This is the number of 4-bit nibbles that are to be either read or written by the port in the buffer.
3:0-35	RESERVED	This word is reserved for future use.

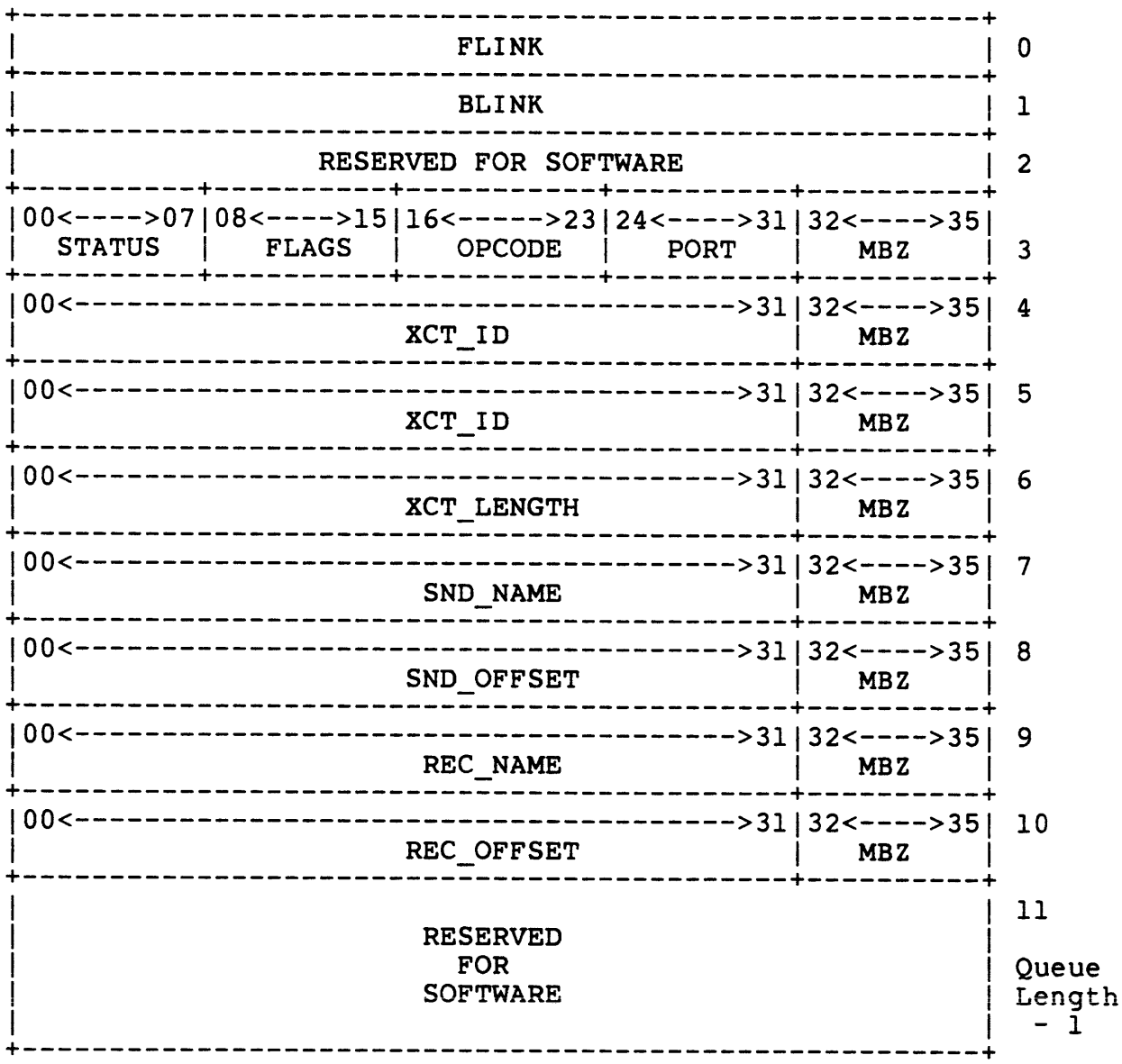
BUFFER SEGMENT DESCRIPTOR FIELD DEFINITIONS

WORD:BITS	NAME	DESCRIPTION
0:0-5	MBZ	This field absolutely MUST be zero; if this field is not zero, correct operation of the port cannot be guaranteed.
0:6-7	MODE	This is the data packing mode for this BSD. 0 => Industry Compatible 1 => Core Dump 2 => High Density 3 => Illegal
0:8-11	MBZ	These bits must be zero.
0:12-35	BSD BASE ADDR	This field contains the address of the start of a physically contiguous block of host memory. This address is the first word of the data buffer.
1:12-35	NEXT BSD	This is the physical address of the first word of the next BSD. If there are no more BSDs, this entire word must be zero.
2:0-35	LENGTH	This is the number of 4-bit nibbles that are to be either read or written by the port in this segment of the buffer.
3:0-35	RESERVED	This word is reserved for future use.

9.2 Reading Data

To read data from a remote host, a host places a special request packet on the port's command queue. This command packet, called a request data packet, (REQDAT), contains the buffer names and offsets of the sending and receiving buffers.

The format of the Request Data (REQDAT) packet is:



WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 10 octal for REQDAT0 11 octal for REQDAT1 12 octal for REQDAT2

4-5:0-31	XCT_ID	This 64 bit word is for the port driver's use; the port will never modify this word. The contents of this location are sent on the CI wire to the other port.
6:0-31	XCT_LENGTH	This is the number of 8-bit bytes that are to be transferred. The actual transfer may be made up of several data packets; this length, however, is the total length of the transfer.
7:0-31	SND_NAME	This is the buffer name of the originating buffer; the data will be read from this buffer.
8:0-31	SND_OFFSET	This is the number of 8-bit bytes, measured from the first byte of the buffer pointed to by the BHD, that the other port will skip over before sending data bytes to this port. These bytes are skipped according to the mode specified by the MODE bits in the first BSD and any succeeding BSD.
9:0-31	REC_NAME	This is the buffer name that the port will write the data into.
10:0-31	REC_OFFSET	This is number of byte positions to skip in the receiving buffer before the first byte of data transmitted over the CI wire is written.

This causes the port to send a Data Request (DATREQ) packet to the other port. The format of the Data Request (DATREQ) is identical to the Request Data (REQDAT). The data is returned in as many packets as necessary by the sending port. After execution, the REQDAT command is normally placed on the MFree Queue unless:

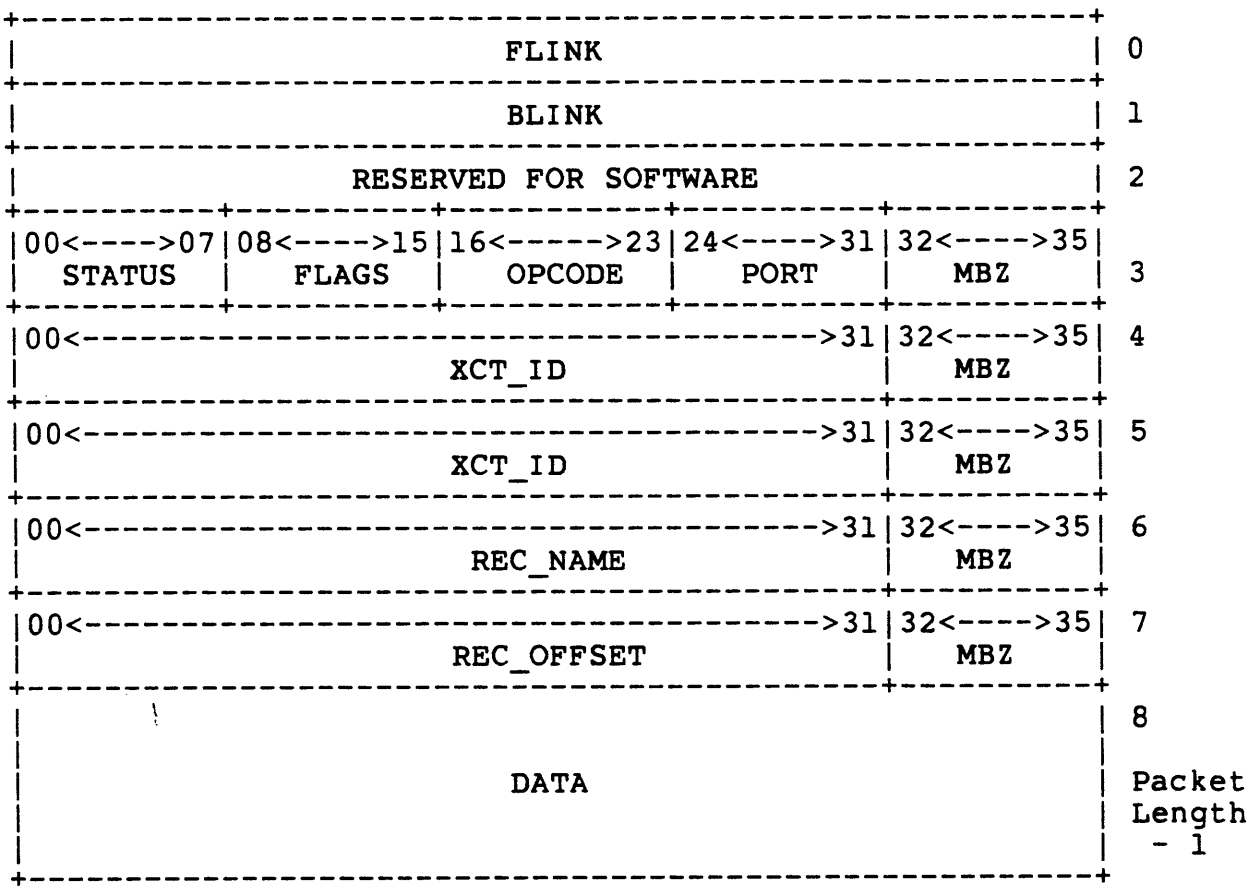
1. The command fails due to an error or the non-receipt of a CI ackowledgment packet.

2. A CI path fails but the command succeeds.

3. The R bit is set in the command.

Upon receipt of a DATREQ packet, the port will build a RETDAT command from an entry taken from the MFree Queue and may place it upon Command Queue 2, 1 or 0 depending upon the opcode of the DATREQ command. The port will put the RETDAT on a command queue only if there is no internal command queue cache available.

The format of the Return Data (RETDAT) packet is:



WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 21 octal for RETDAT

4-5:0-31	XCT_ID	This 64 bit word is for the port driver's use; the port will never modify this word. The contents of this location are sent on the CI wire to the other port.
9:0-31	REC_NAME	This is the buffer name that the port will write the data into.
10:0-31	REC_OFFSET	This is number of byte positions to skip in the receiving buffer before the first byte of data transmitted over the CI wire is written.

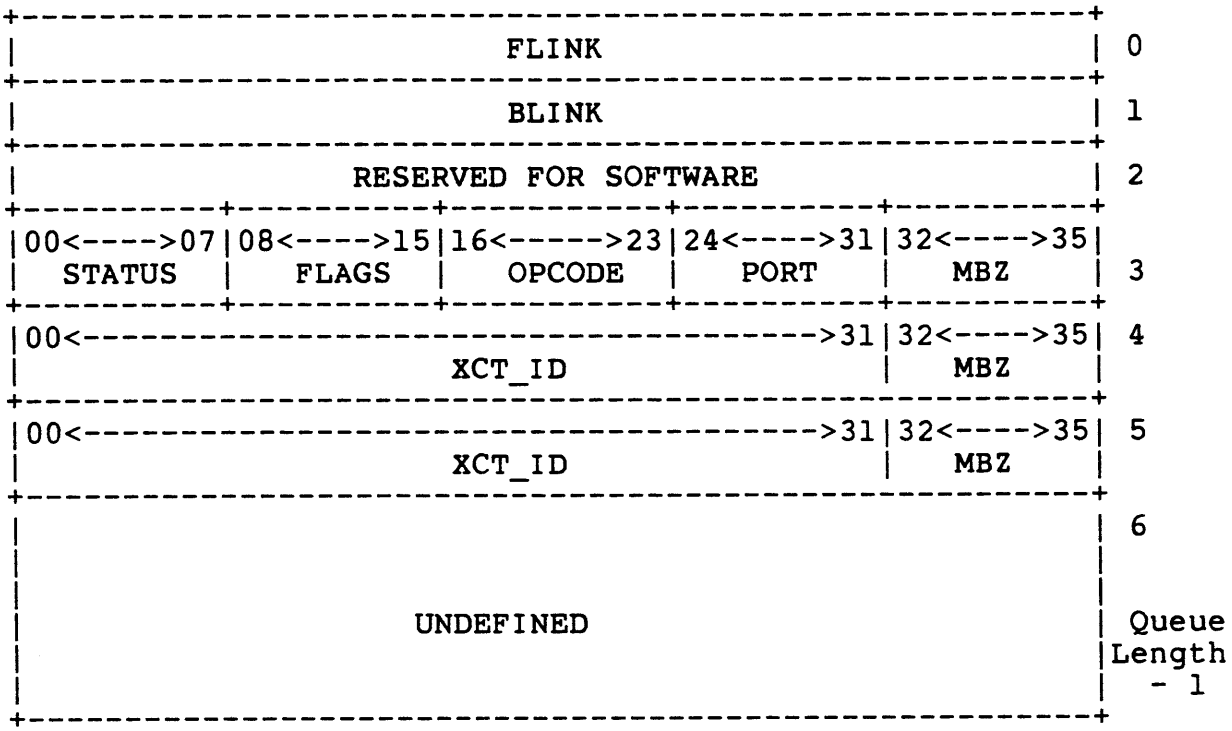
The last data packet of the transfer has the Last Packet (LP) flag set to inform the receiving port that all of the data has been sent. When the RETDAT command has been executed, it will be returned to the MFree Queue unless:

1. The command fails due to an error.

2. A CI path fails but the command succeeds.

In these cases a Data Returned (DATRET) response is generated and placed on the Response Queue by the sending port. The format of a Data Returned (DATRET) response is identical to a Return Data (RETDAT) except that no data appears in the command. The port driver should never build a RETDAT command, but must be able to tolerate finding a RETDAT command that the port placed upon its command queue after receiving a DATREQ packet. The port receiving the data will generate a Data Received (DATREC) response when all of the data has been received (LP flag set) or an error occurs.

The format of the Data Received Response (DATREC) is:



WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 61 octal (DATREC).

9.3 Writing Data

To write data to a remote port, the port driver places a send data (SNDDAT) command upon the port's command queue. The port will then access the BHD and associated BSDs, transmitting as many data packets as necessary to move the data from the sending buffer to the receiving buffer's port, as designated in the port field of the command. BHD's and BSD's are discussed earlier. This operation takes place under Virtual Circuit control; any error causes the circuit to be closed. After execution, the SNDDAT queue entry is normally inserted on the MFree Queue but it will be placed on the Response Queue as a Data Sent (DATSNT) response if one of the following conditions occurs:

1. The command fails due to an error or non-receipt of a packet acknowledgement. This closes the Virtual Circuit.

2. A CI path fails but the command succeeds on a retry.

3. The R bit of the command is set.

The format of a Send Data (SNDDAT) and a Data Sent (DATSNT) are identical to a Request Data (REQDAT) command except that the opcode is 20 octal.

As data is received, the data is stored by the receiving port. Upon receipt of a Sent Data packet (SNTDAT) with the last packet flag set, the receiving port will:

1. Remove an entry from the MFree Queue.

2. Generate a Return Confirm (RETCNF) command.

3. Insert the RETCNF command on Command Queue 3.

When executed a RETCNF sends a confirmation packet to the initiating port. After execution the RETCNF command is normally placed on the MFree Queue but it will be placed upon the Response Queue as a Confirm Returned (CNFRET) Response if:

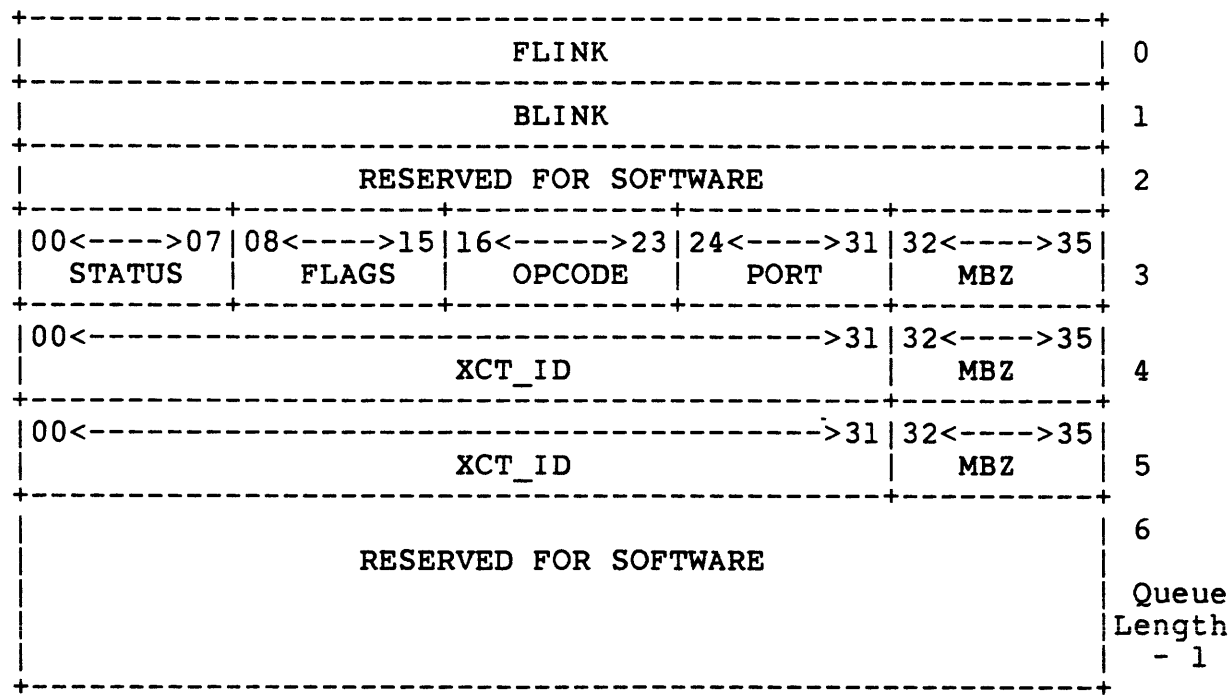
1. The command fails due to an error or non-receipt of a CI acknowledgement packet.

2. A CI path fails but the command succeeds.

If any data packet can not be properly processed by the receiving port, it will:

1. Close the Virtual Circuit.
2. Remove an entry from the MFree Queue.
3. Build a Confirm Received (CNFREC) response and insert it on the Response Queue.

The format of the response queue entry for a CNFRET packet is:



WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 3 octal (CNFRET). 43 octal (CNFREC).

4-5:0-31	XCT_ID	This is the transaction ID that accompanied all of the data packets. This information is for the use of the port driver.

9.4 Data Formatting Modes

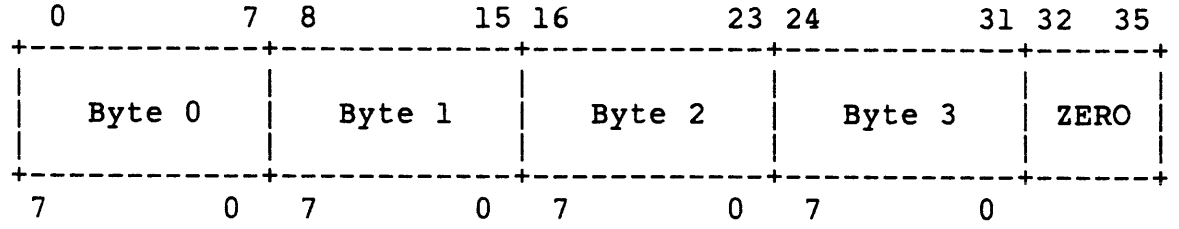
The KL10 CI port supports 3 data formatting modes. These modes allow the operating system to specify how the data is packed in the host's memory on memory read operations and how the port should write the data into the host's memory on write operations. These modes are specified in each Buffer Segment Descriptor for data transfers via named buffers.The three modes are Industry Compatible, High Density, and Core Dump. The first two of these modes may also be specified for datagram and message transfers through the various Send Message and Send Datagram commands. 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.

9.4.1 Industry Compatible Mode -

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

INDUSTRY COMPATIBLE

KL10 WORD



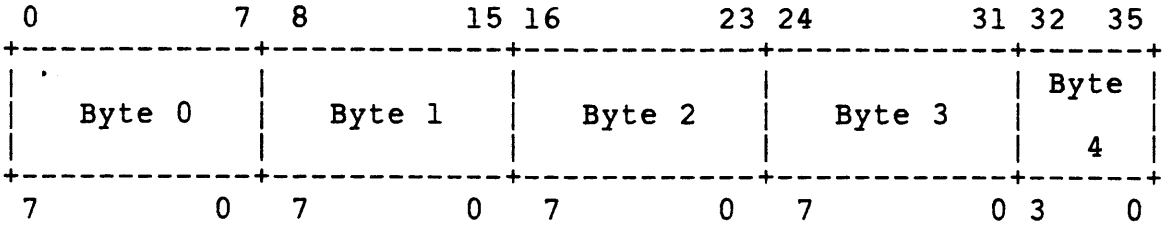
PLI BYTE

9.4.2 Core Dump Mode -

The following figure illustrates the core dump mode for mapping 8 bit CI bytes into 36 bit KL10 words. It is important to note that in core dump mode the high order 4 bits (PLI bits 7 to 4) of the last byte of each 36 bit word are written to the CI as zeroes and are read from the CI, but are discarded by the port. Core Dump BHD'S and BSD's are expected to be in multiples of 10. even though only 9. are moved for each 36 bit word.

CORE DUMP

KL10 WORD



PLI BYTE

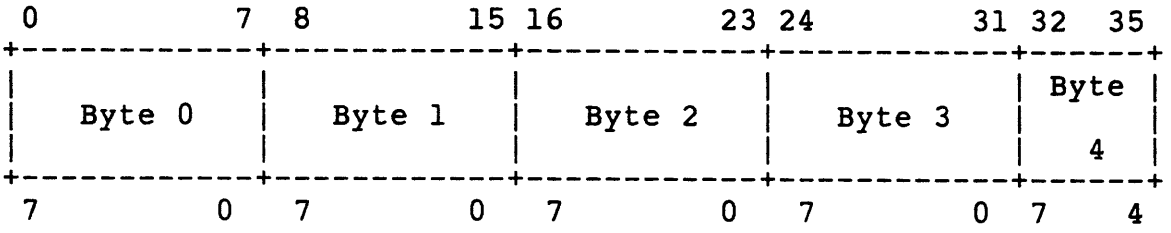
9.4.3 High Density Mode -

The following figure illustrates the high density mode for mapping 8 bit CI 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

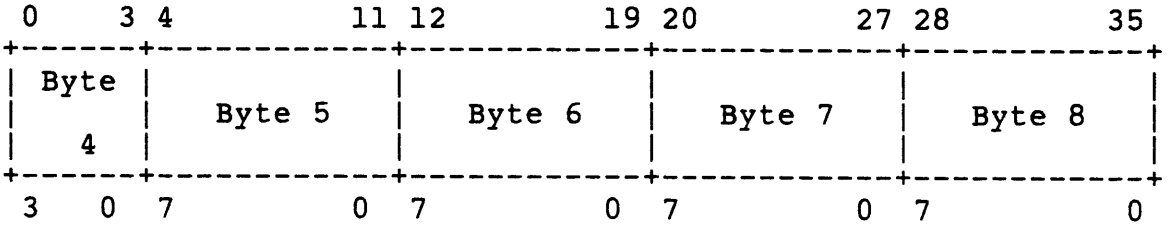
KL10 WORD PAIR

FIRST WORD OF PAIR



PLI BYTE

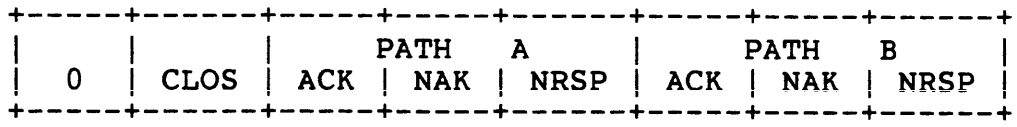
SECOND WORD OF PAIR



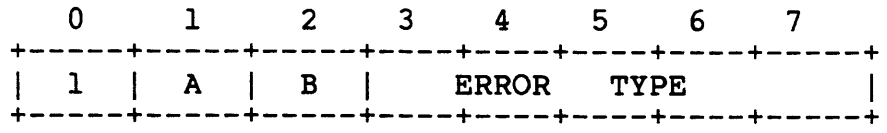
PLI BYTE

10.0 STATUS FIELD

The STATUS field is updated by the port when it builds a response queue entry. The various valid values of the STATUS field are defined below. Note that bit 0 of the STATUS field defines the definition of the remaining bits.



BIT	NAME	DESCRIPTION
1	CLOS	A packet had a retry failure on a path but was transmitted successfully on the other path. The path that failed and the type of failure is indicated in the Path bits. The indicated path is also marked as being bad in the VCDT (Virtual Circuit Descriptor Table).
2	PATH A ACK	The packet was ACKed on this path.
3	PATH A NAK	The packet was NAKed at least once on this path.
4	PATH A NRSP	The packet received No ReSPonse at least once on this path.
5	PATH B ACK	The packet was ACKed on this path.
6	PATH B NAK	The packet was NAKed at least once on this path.
7	PATH B NRSP	The packet received No ReSPonse at least once on this path.



BITS	NAME	DESCRIPTION
1	PTH_A	The error is associated with path A.
2	PTH_B	The error is associated with path B.
3-7	ERROR TYPE	

NO PATH ERRORS

(402) ERROR = 1 => Access Control violation.
(404) ERROR = 2 => Invalid Buffer Name.
(406) ERROR = 3 => Buffer Length violation.
(410) ERROR = 4 => Packet size violation.
(412) ERROR = 5 => Transmit Buffer Parity Error
(414) ERROR = 6 => Local unrecognized command.
(416) ERROR = 7 => Mover Parity Predictor Error.
(420) ERROR = 10 => Invalid Remote port.
(422) ERROR = 11 => Non-recoverable Long Word Count error
(424) ERROR = 12 => No legal path.
(426) ERROR = 13 => Command not legal in disabled state.
(430) ERROR = 14 => PLI data PE in SRC byte.
(432) ERROR = 15 => PLI data PE in OPC byte.
(434) ERROR = 16 => PLI data PE in body.
(436) ERROR = 17 => Port disabled during processing.
(440) ERROR = 20 => Xmit Buffer Parity Error.
(442) ERROR = 21 => Channel Error.
(444) ERROR = 22 => Spurious Channel Error.

Path B Errors

(502) ERROR = 41 => Remote unrecognized command
(504) ERROR = 42 => Virtual Cicuit closed
(506) ERROR = 43 => Retries Exhausted (NAK)
(510) ERROR = 44 => Retries Exhausted (NRSP)
(512) ERROR = 45 => Transmitter Timeout

PATH A Errors
=============

(602) ERROR = 101 => Remote unrecognized command
(604) ERROR = 102 => Virtual Cicuit closed
(606) ERROR = 103 => Retries Exhausted (NAK)
(610) ERROR = 104 => Retries Exhausted (NRSP)
(612) ERROR = 105 => Transmitter Timeout

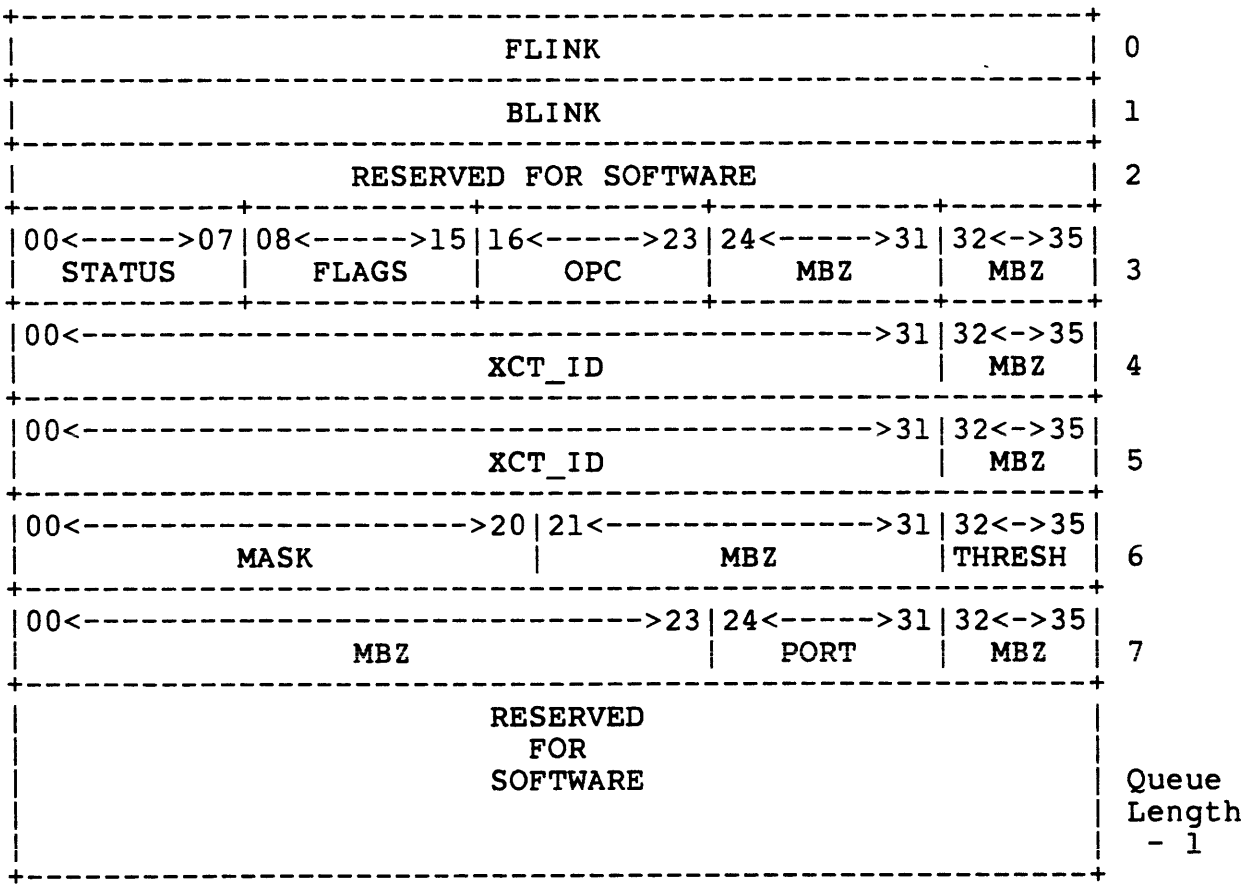
PATHS A,B Errors
================

(704) ERROR = 142 => Virtual Cicuit closed
(706) ERROR = 143 => Retries Exhausted (NAK)
(710) ERROR = 144 => Retries Exhausted (NRSP)
(712) ERROR = 145 => Transmitter Timeout

11.0 PORT PERFORMANCE MONITORING

The port microcode implements several counters which are under the control of the port driver. The command queue entry Set Counters (SETCNT) allows the port driver to point and/or clear the counters. It also allows the port driver to enable or disable the event counting. There is a mask that is used to control the loading and enabling of the various event counters. For each counter, there are 2 bits in the mask; the first bit enables the counting of the event, and the second bit controls the clearing of the event counter. The port driver may instruct the port to count events for a specified port or a cumulative count for all ports.

The format of the SETCNT command is:



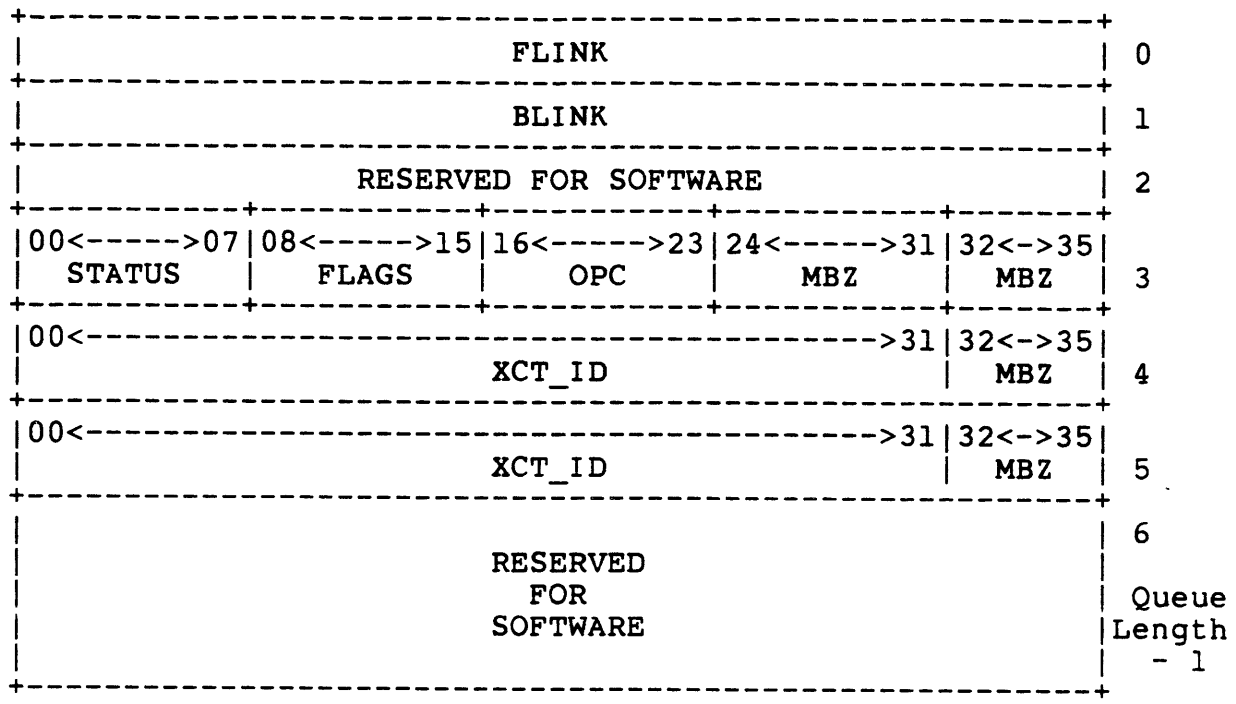
WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 201 octal (SETCNT).
6:0-19	MASK	This is the 18 bit mask used to control the enabling and loading of the counters.
6:0	PTH_A ACK	If on, count ACKs received on Path A.
6:1	PTH_A ACKC	If on, clear the counter.
6:2	PTH_A NAK	If on, count NAKs received on Path A.
6:3	PTH_A NAKC	If on, clear the counter.
6:4	PTH_A NRSP	If on, count NO_RSPs received on Path A.
6:5	PTH_A NRSPC	If on, clear the counter.
6:6	PTH_B ACK	If on, count ACKs received on Path B.
6:7	PTH_B ACKC	If on, clear the counter.
6:8	PTH_B NAK	If on, count NAKs received on Path B.
6:9	PTH_B NAKC	If on, clear the counter.
6:10	PTH_B NRSP	If on, count NO_RSPs received on Path B.
6:11	PTH_B NRSPC	If on, clear the counter.
6:12	DG DISCARDED	The count of discarded datagrams because of no DGFree Queue entries.
6:13	DG DISC CLR	If on, clear the counter.
6:14	XMT CNT	Count the packets transmitted to the designated port.
6:15	XMT CLR	If on, clear the counter.
6:16	RCV CNT	Count the packets received from the designated port.

6:17	RCV CLR	If on, clear the counter.
6:18	ERR_CNTR_CLR	If on, clear all error counters (see CNTRD response).
6:19	SET_THRESH	If on, load Port Recoverable Error Threshold value.
6:20	NO_ANSWER	If on, don't answer received IDREQ with ID. This effectively puts the port in a Maintenance state. This bit is valid only if the port is in the DISABLED state.
6:32-35	THRESH_VAL	Value to load for Port Recoverable Error Threshold.
7:24-31	PORT	This is the designated port for which the above counters will be tracked. If the port value is set to 255, then the counting will be done for all ports.

If the R (response) bit is set in the Set Counters command (SETCNT) it will be placed on the Response Queue instead of the DGFree Queue as a counters Set (CNTSET) command. The format for a Counters Set (CNTSET) command is identical to the Set Counters (SETCNT) command.

Every time the port enters the Enabled state, it will clear all of the counters and set the PORT field to the "all ports" value. The port driver reads these counters, with a Read Counters (RDCNT) command. This command will return the information in the various counters.

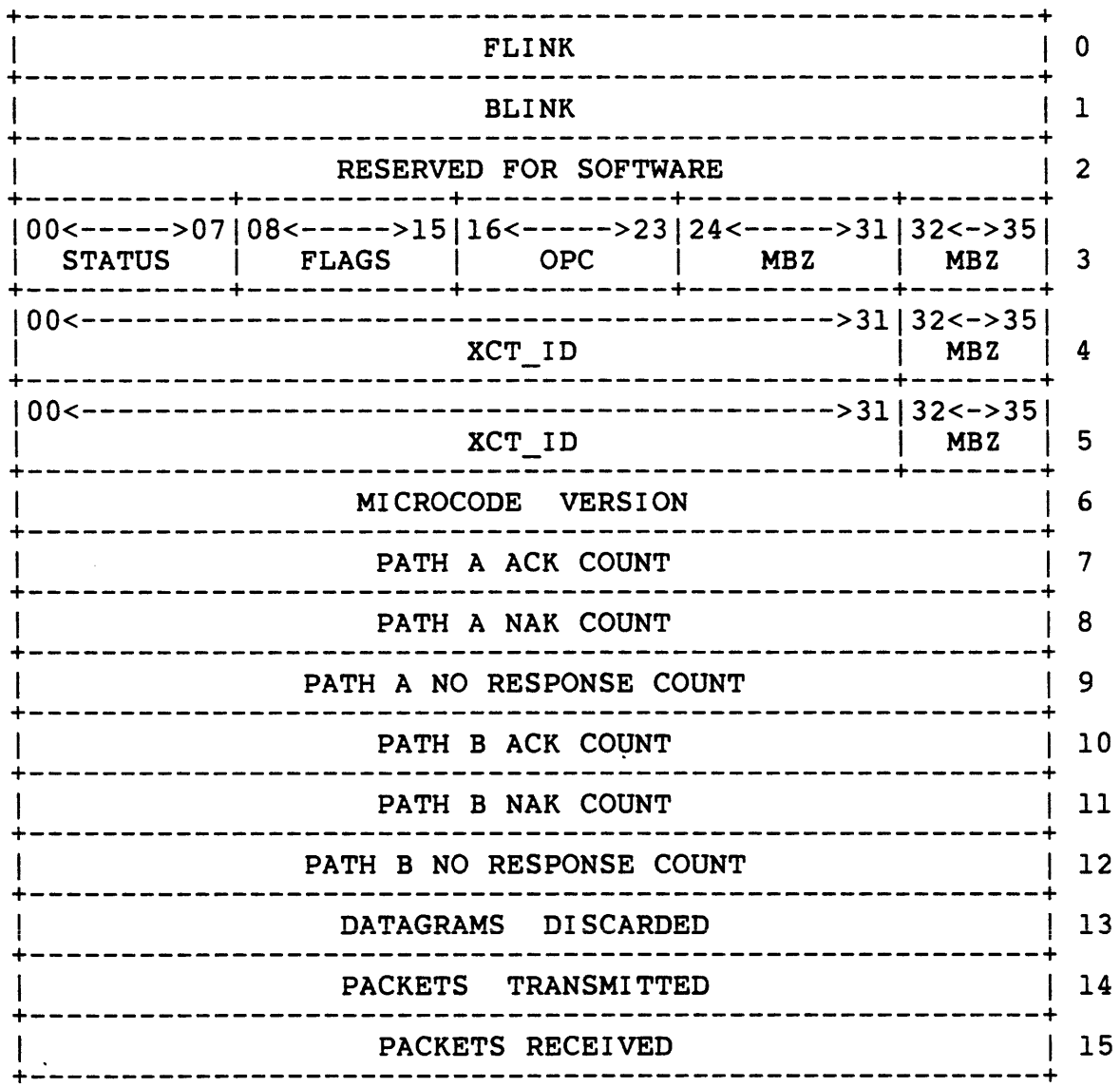
The format of a Read Counters (RDCNT) command is:

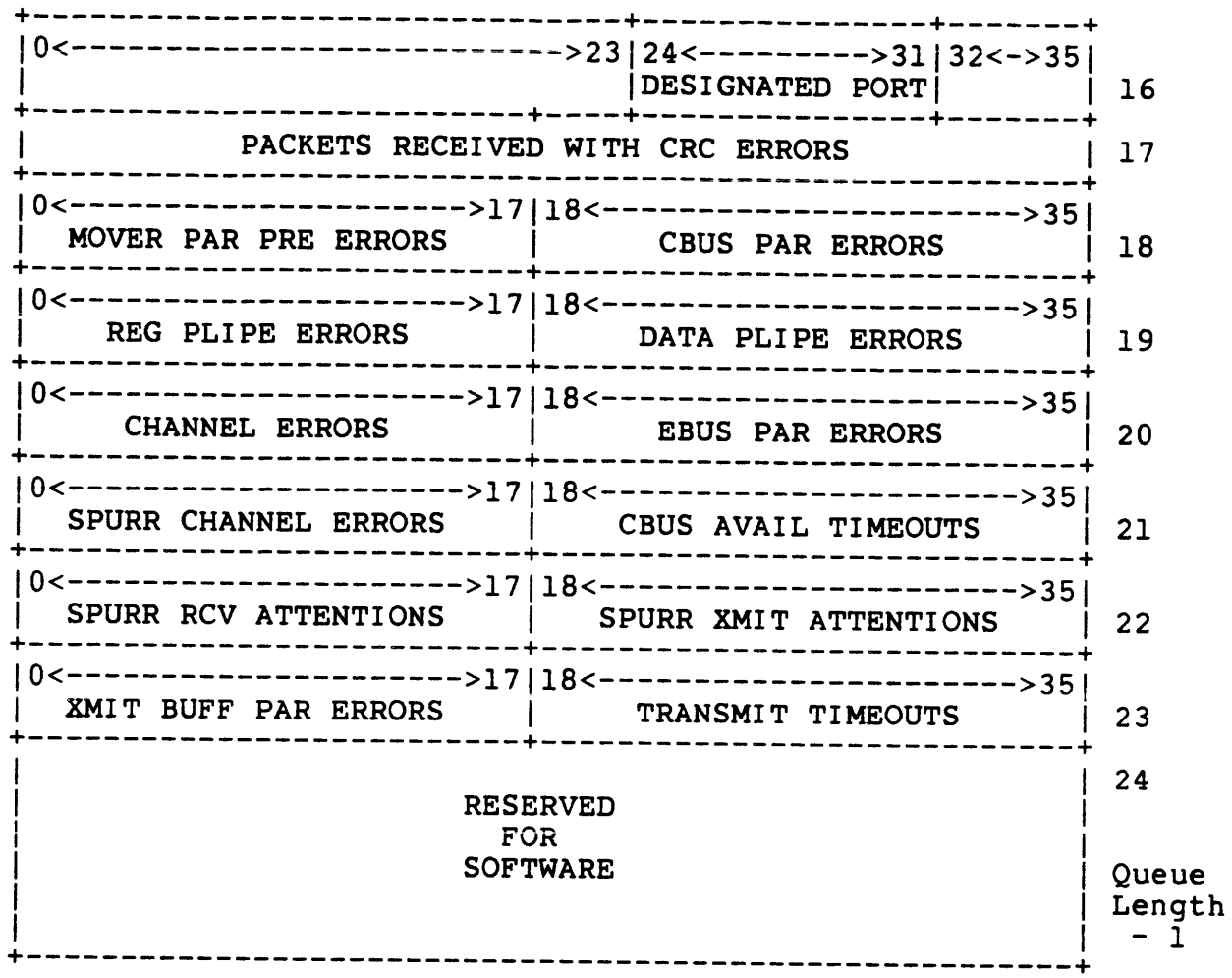


WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 202 octal (RDCNT).

The port will always generate a Counters Read (CNTRD) response to the Read Counters (RDCNT) command.

The format of the Counters Read (CNTRD) response is :





WORD:BITS	NAME	DESCRIPTION
3:12	ERROR	This bit is set if the CNTRD was generated as a result of a Planned CRAM Parity Error (see KLCI Error spec).
3:16-23	OPCODE	OPCODE = 202 octal (CNTRD).

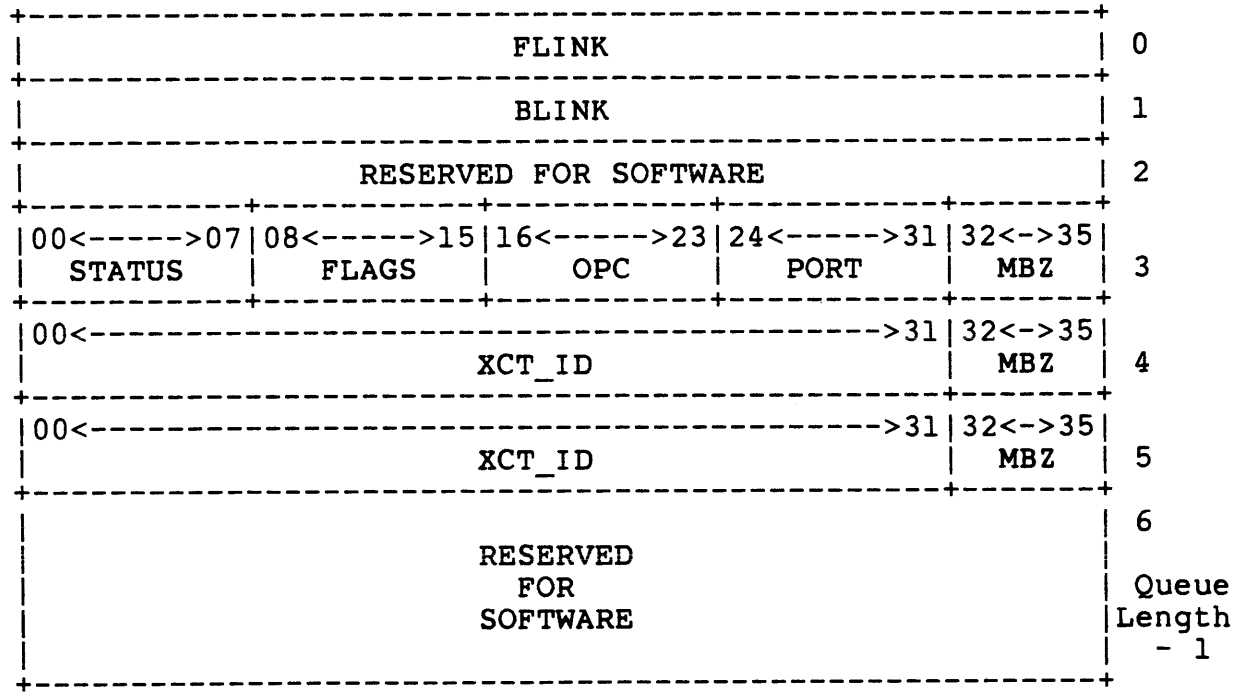
Words 18-23 are called the Port Recoverable Error Counters. The errors have a threshold initially set to 5 by the port during initialization. The threshold can be changed by the port driver with the SETCNT command. The threshold has a value range of 0-17.

12.0 MISCELLANEOUS MESSAGES

12.1 Configuration Information

There is also a class of commands called ID commands. The Request ID (REQID) command enables a host to determine the configuration of the CI bus. This command generates an ID Request (IDREQ) packet that is sent to the appropriate CI port. The receiving port responds with a Send ID (SNDID) packet. A command is taken from the DGFree Queue and placed on the Response Queue as a ID Received (IDREC) command. All ID packets are datagram-class packets. It is illegal for the port driver to place a Send ID (SNDID) command on the command queue; SNDID packets are generated by the port microcode only. When the port is in the Enabled state, it will automatically respond to REQID packets without informing the port driver.

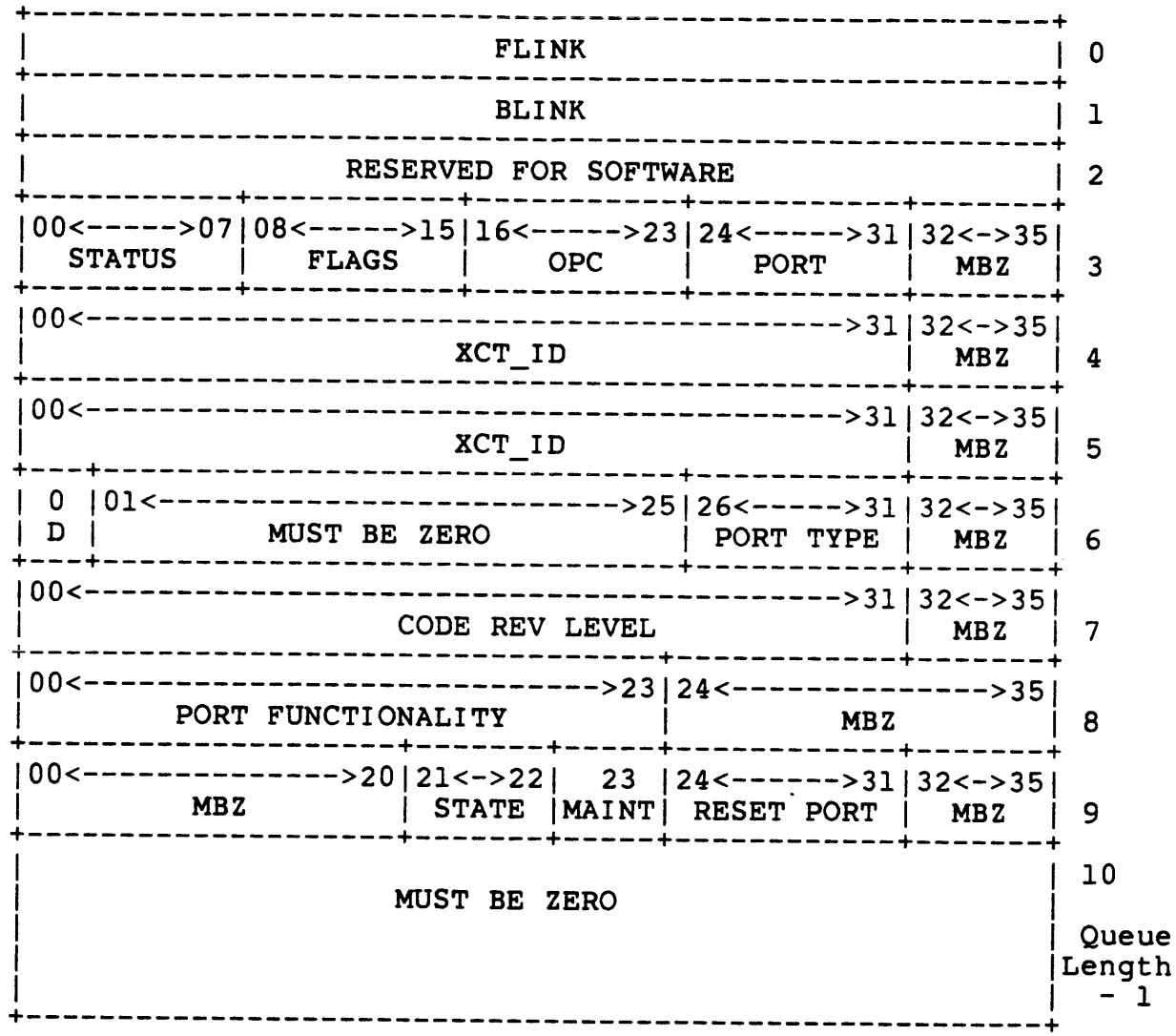
The format of the REQID command and the IDREQ response is:



WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 5 octal (REQID)(IDREQ).

4-5:0-31 XCT_ID Transaction ID for software.

The format of the ID Received (IDREC) response is:



WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 53 octal (IDREC).
4-5:0-31	XCT_ID	Transaction ID for software.

6:0	DUAL	The responding port has two paths.
6:1-25	MBZ	Must be zero.
6:26-31	PORT TYPE	This is the port type. (6 = KL10)
6:32-35	MBZ	Must be zero.
7:0-31	CODE REV	This is the microcode edit level.
8:0-23	FUNCTIONALITY	This is a bit mask describing the functionality of the originating port. Refer to the corporate CI spec for the bit definitions.
9:21-22	PORT STATE	0 => Unitialized 1 => Disabled 2 => Enabled
9:23	MAINTENANCE	The port is in the maintenance state. 0 => not in maintenance mode 1 => in maintenance mode
9:24-31	RESET PORT	This is the port number of the last port which sent the port a reset packet. This will always be our port number.

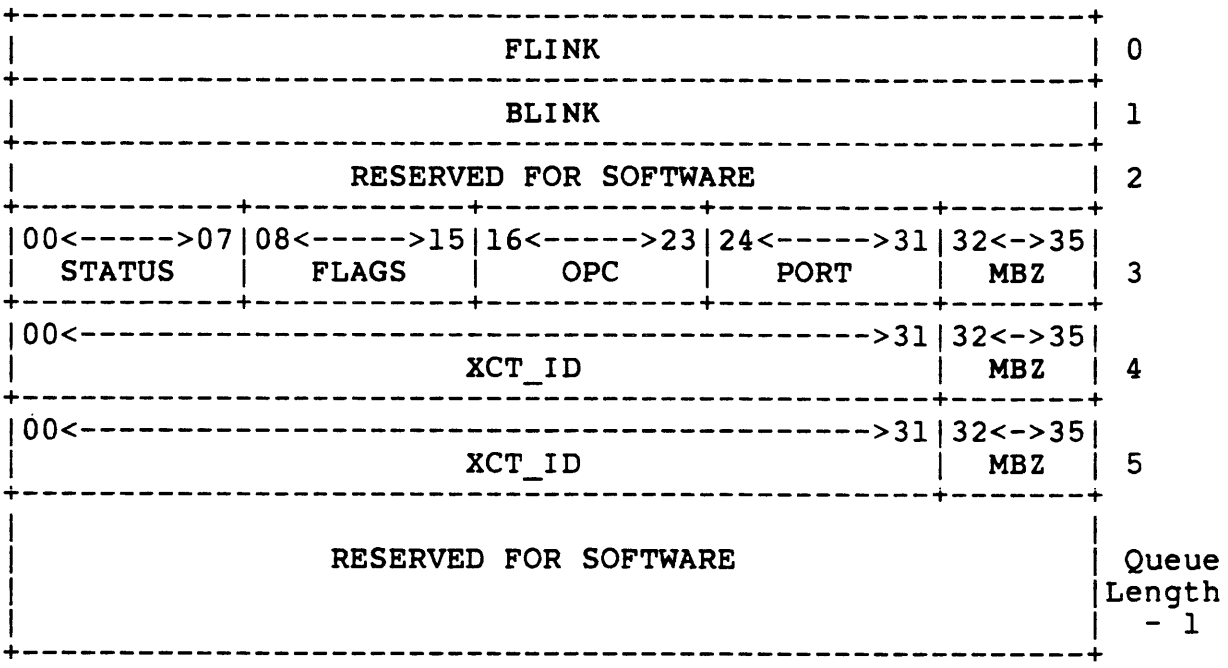
12.2 Maintenance Commands

The LCG port supports the CI maintenance packets for the purpose of loading, starting, reading, and writing of other hosts' memories. All maintenance packets on the CI are given datagram-class service. Since the LCG CI port is not down-line loadable and since a remote port may not start or stop LCG host computers, all those maintenance packets are passed to the port driver via the Response Queue when they are received over the CI wire. Maintenance class commands are Reset (SNDRST), Start (SNDSTRT), Request Maintenance Data (REQMDAT), Send Maintenance Data (SNDMDAT), and Maintenance Confirm Return (MCNFRET). Because the LCG CI port does not have a Maintenence state all Maintenence class commands and responses are always valid.

12.2.1 Resetting Remote Systems -

The Send Reset (SNDRST) command is used to reset a nodes' interface and host to a known state (Unitialized/Maintenance) and to halt the host computers' execution. See the Computer Interconnect Spec for an explaination of port states.

The format of the Send Reset (SNDRST) command and Reset Sent (RSTSNT) and Restart Received (RSTREC) responses is:

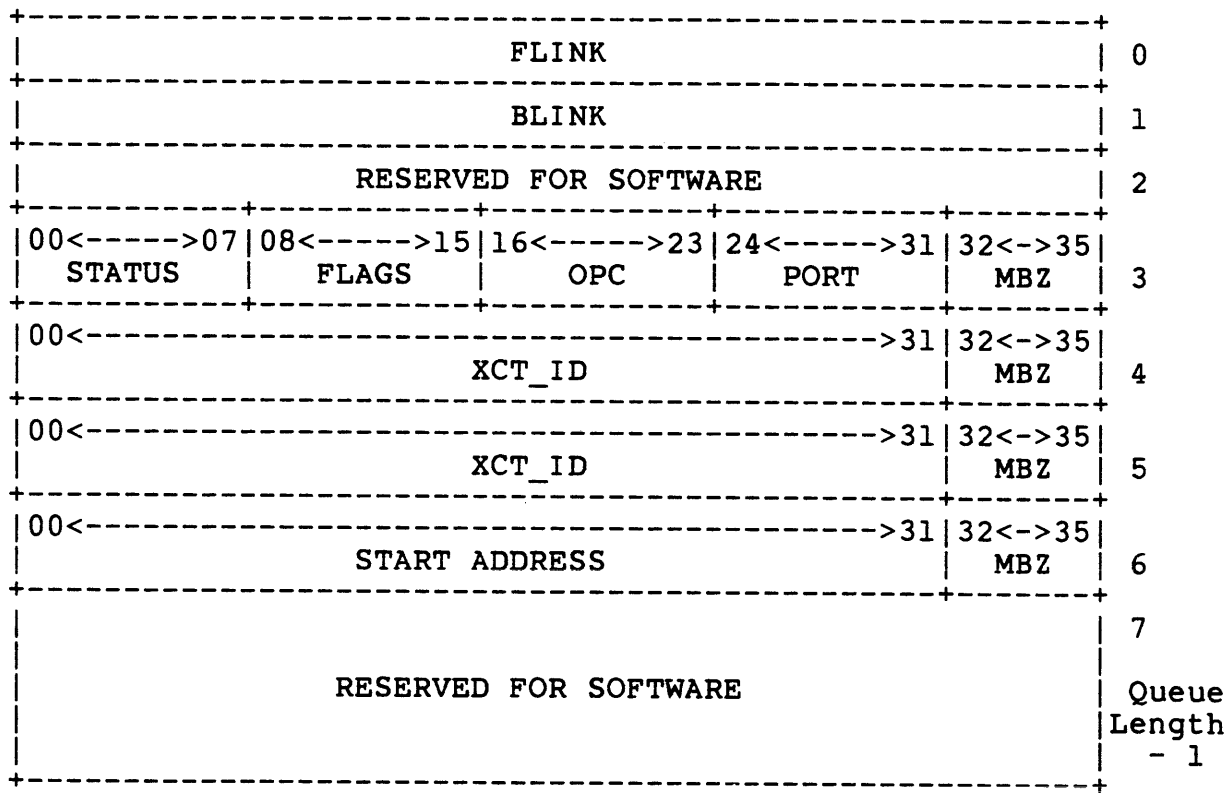


WORD:BITS	NAME	DESCRIPTION
3:8	FORCE	If this bit is off, the port will compare the source port number to the port number stored in the last resetting port number. If there is a match, the reset occurs, otherwise the packet is ignored. If the bit is on, the port will perform an unconditional reset.
3:16-23	OPCODE	OPCODE = 6 octal (SNDRST)(RSTSNT). 46 octal (RSTREC)

12.2.2 Starting Remote Systems -

A Remote System Start (SNDSTRT) command is sent to ports that have been stopped by a SNDRST command. The start function will be performed only if the RSTPORT field of the receiveing port is equal to the source port number. A node may be started only by the port that stopped it.

The format of the Send Start (SNDSTRT) command and Start Sent (STRTSNT) and Start Received (STRTREC) responses is:



WORD:BITS	NAME	DESCRIPTION
3:8	USE DSA	If this bit is set, the remote host should use the default starting address.
3:16-23	OPCODE	OPCODE = 7 octal (SNDSTRT)(STRTSNT) 47 octal (STRTREC)

5-6:0-31	START ADDRESS	This is the starting address if the Default Starting Address bit is not on.

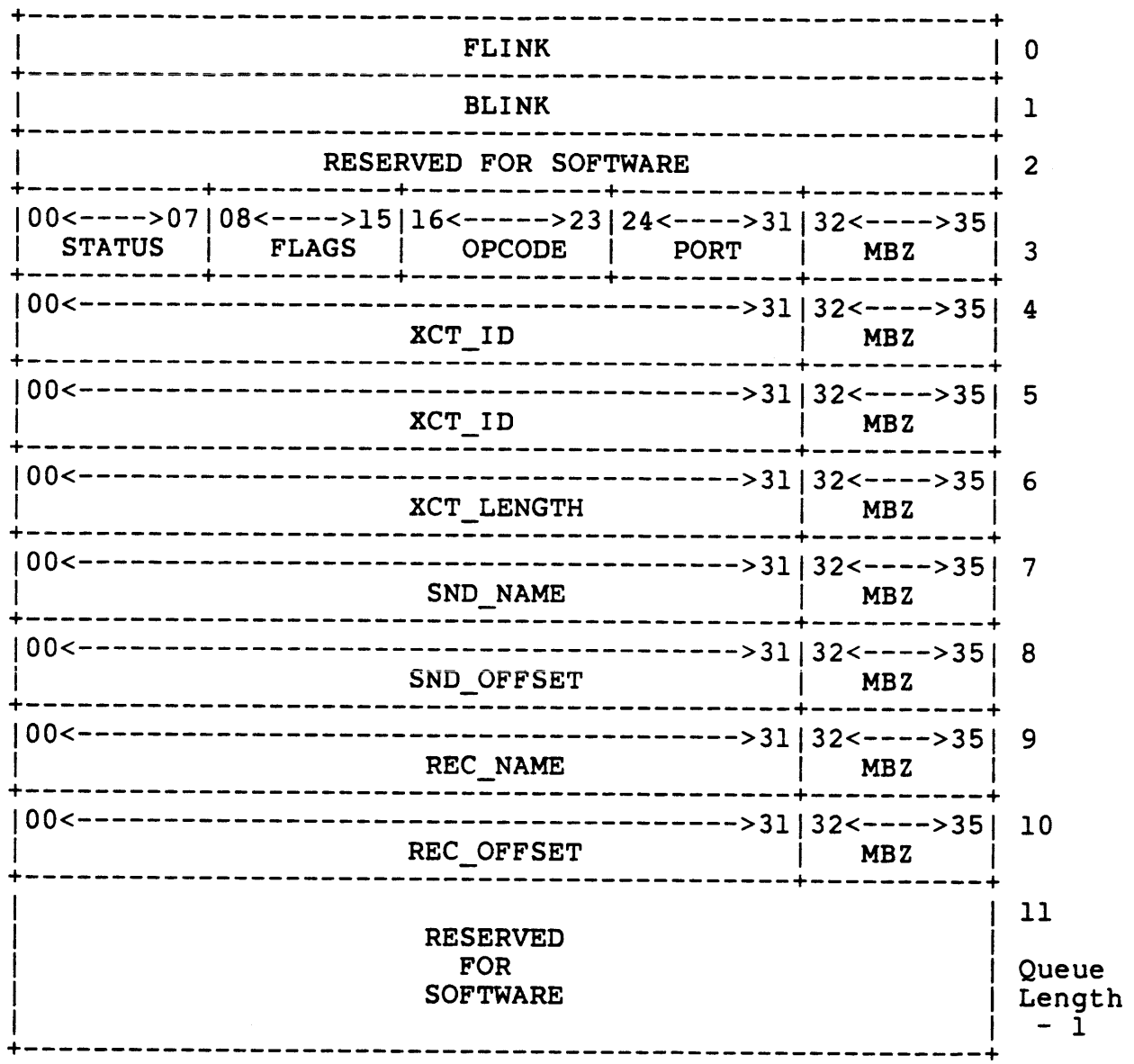
12.2.3 Reading Remote System Memories -

Remote host system memory may be read via the Request Maintenance Data (REQMDAT) packet. Data read in this manner is returned via the Returned Maintenance Data (RETMDAT) packet. The resulting data will appear in the specified buffer. The receiving buffer must have a valid buffer name and the BHD and BSDs must be properly set up as if it were a normal data transfer. Since the data transfer takes place as a datagram, a Virtual Circuit is not necessary. Whenever a maintenance data transfer is complete, the port microcode will build a Maintenance Confirm response on the response queue. The format of the REQMDAT command queue entry is the same as a REQDAT command except that the opcode is 16.

12.2.4 Writing Remote System Memories -

The Send Maintenance Data (SNDMDAT) command packet is used to write a remote host's memory. The length of the data to be sent is no greater than 512 bytes or 576 bytes, according to the Packing Size bit in the FLAGS field.

The format of the SNDMDAT command queue entry is:

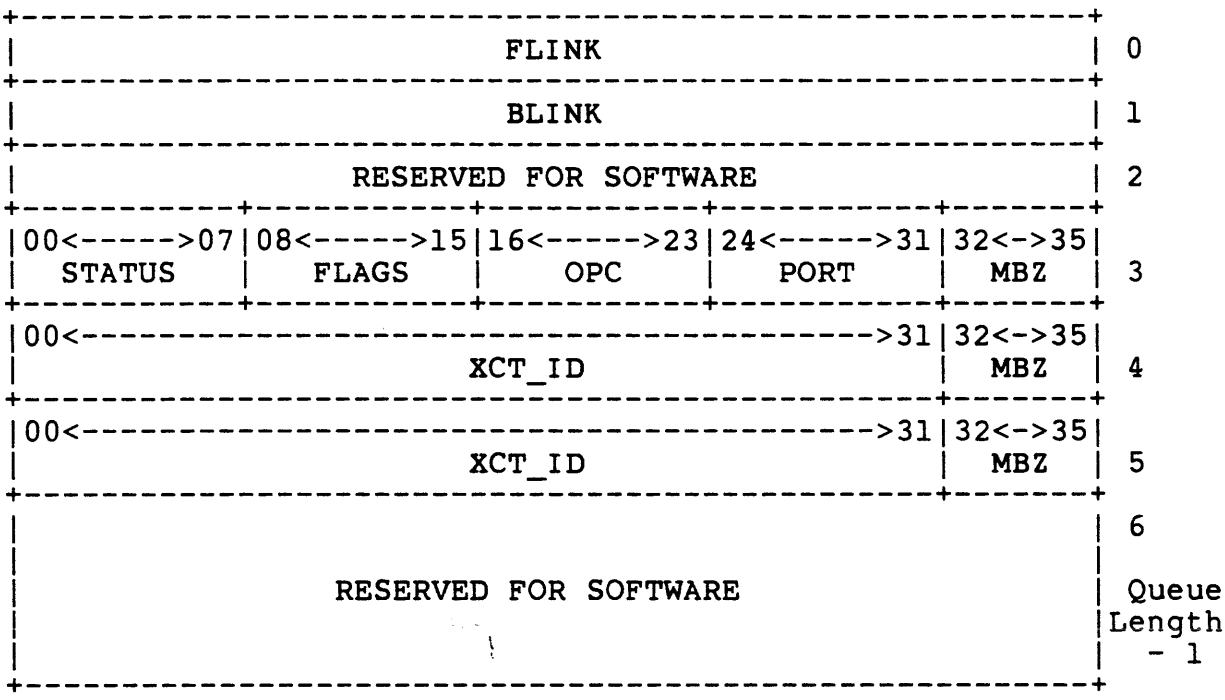


WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 22 octal (SNTMDAT).

12.2.5 Maintenance Confirmation Packets -

Whenever a maintenance data read or write is completely successfully at the remote port, a Maintenance Confirm packet will be sent back to the originating port. The port microcode will place this confirmation packet on the tail of the response queue.

The format of the MCNF queue entry is:



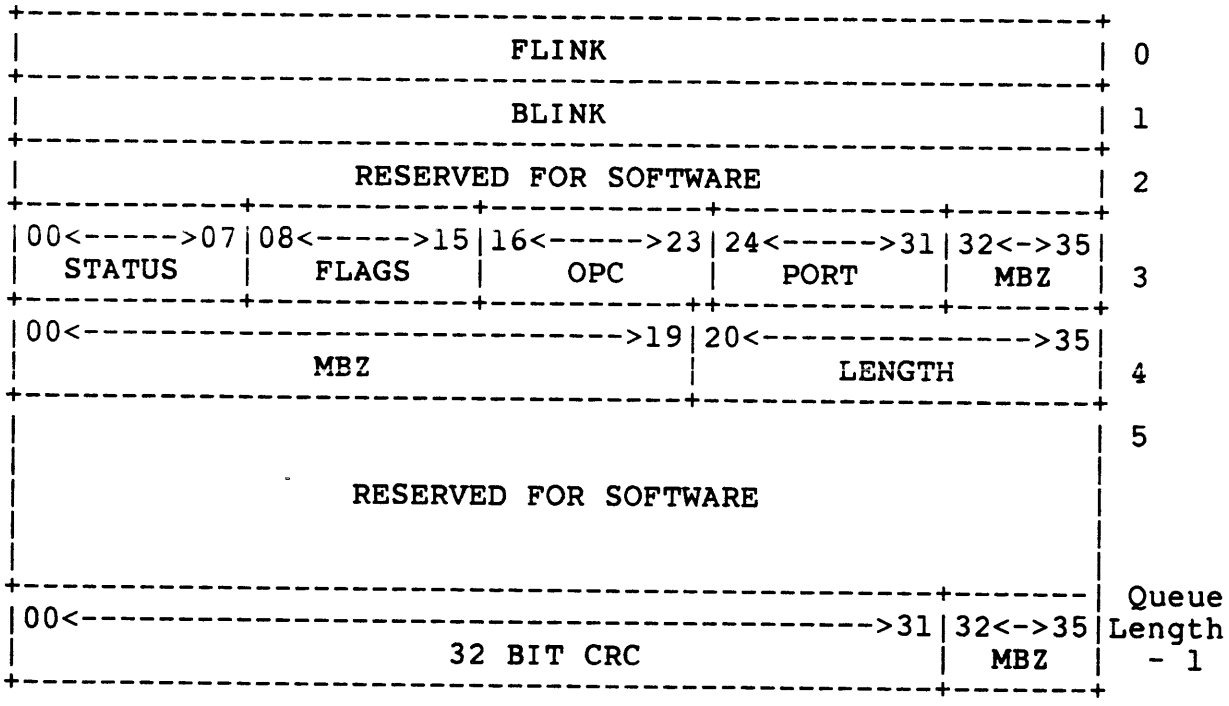
WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 44 octal (MCNF)
4-5:0-31	XCT_ID	This is the transaction ID specified in the original maintenance read or write memory command.

12.3 Diagnostic Operation

12.3.1 Loopback -

There is a special diagnostic command called Send Loopback (SNDLB). This command is a Datagram class command. This command along with the IDREQ command are used by ports to verify and isolate faults in cluster path connections. When a Loopback Received (LBREC) packet is received it is placed on the Response Queue even if it is from a port other that itself.

The format of the Send Loopback (SNDLB) command and Loopback Sent (LBSNT) and Loopback received (LBREC) responses is:



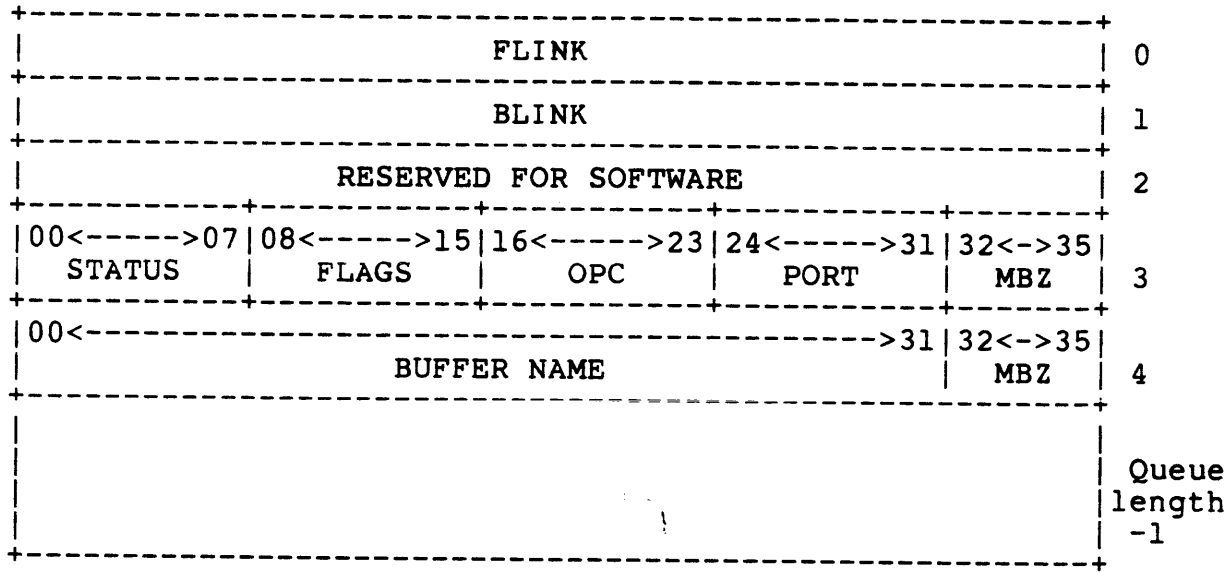
WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 15 octal (SNDLB).

4:20-35	LENGTH	This is the number of bytes, not including the 4 bytes of CRC, that are in the text portion of the LOOPBACK packet.
??:0-31	32-BIT CRC	The software must place the correct 32-bit CRC polynomial as calculated via the algorithm in the VAX11/780 Architecture Handbook, page 9-13.

12.4 Data Buffer Maintenance

The operating system requires the ability of flushing any information cached by the port concerning data buffers that require closing. The Close Buffer (CLSBUF) command forces the port to close the data buffer named by clearing the "V" bit in the BHD and flush any commands that are using that buffer. The commands that are using the buffer and any subsequent command or packet will be placed on the response queue with the STATUS field indicating an error.

The format of the Close Buffer (CLSBUF) command and the Buffer Closed response (BUFCLS) is:



WORDS:BITES	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 205 octal (CLSBUF)(BUFCLS)
4	BUFNAM	This is the name of the buffer to be closed. The buffer name is the same format as is used in named buffer transfers. See section 9.1 for a description of buffer names.

12.5 Illegal Packets

Any packet which is processed while the port is in the enabled state and is an illegal packet due to either the opcode or some other field will be treated as a datagram and placed on the host's response queue. If the packet was locally generated, then the port microcode will place the response on the tail of the response queue with the STATUS field indicating a local unrecognized command. If the packet was received over the CI wire, the STATUS field will be set to remote unrecognized command. The entire contents of the packet, up to the specified queue entry length, will be placed on the response queue. Since this packet is considered to be a datagram, it is subject to being discarded if there is a lack of buffer space in the port. Also note that the packet length field will be filled in by the port just as if it were a regular datagram packet. The opcode will be unchanged.

13.0 PORT REGISTERS

13.1 Control And Status Register (CSR)

The Control and Status Register (CSR) is the mechanism used by the LCG CI port and it's host to communicate with each other. The CSR is described in a later section and in the CI20 Port Architecture Spec.

BIT NO.	BIT DEFINITION	RD/WR KL10	RD/WR PORT
00	PORT PRESENT	R	H
01	DIAG RQST CSR	R	H
02	DIAG CSR CHNG	R/H	H
03		*	*
04	RQST EXAM OR DEP	R/H	R/S
05	RQST INTERRUPT	R/H	R/S
06	CRAM PARITY ERR	R/C	H
07	MBUS ERROR	R	H
08		*	*
09		*	*
10		*	*
11	IDLE LOOP	R	R/W
12	DISABLE COMPLETE	R	R/W
13	ENABLE COMPLETE	R	R/W
14		*	*

BIT NO.	BIT DEFINITION	RD/WR KL10	RD/WR PORT
18	CLEAR PORT	W	*
19	DIAG TEST EBUF	R/W	*
20	DIAG GEN EBUS PE	R/W	*
21	DIAG SEL LAR	R/W	*
22	DIAG SINGLE CYC	R/W	*
23	SPARE	R/W	*
24	EBUS PARITY ERR	H/R/C	R
25	FREE QUEUE ERR	R/C	R/S
26	DATA PATH ERR	R/C	R/S
27	CMD QUEUE AVAIL	R/S	R/C
28	RSP QUEUE AVAIL	R/C	R/S
29		*	*
30	DISABLE	R/S	R/C
31	ENABLE	R/S	R/C
32	MPROC RUN	R/W	R/H

15	PORT ID CODE 00	R	H	33	PIA 00	R/W	R
16	PORT ID CODE 01	R	H	34	PIA 01	R/W	R
17	PORT ID CODE 02	R	H	35	PIA 02	R/W	R

"*" indicates that the bit is not defined
"R" indicates that the bit is readable
"W" indicates that the bit is writeable
"C" indicates that the bit may be cleared only as a single bit
"S" indicates that the bit may be set only
"H" indicates that the bit is hardware controlled

13.2 Diagnostic Registers

In order to support the diagnosis of the port, all of the packet buffer board and link registers are accessable to a diagnostic program. These registers are accessed via two local commands, the Read Register (RDREG) and Write Register (WRREG) commands.

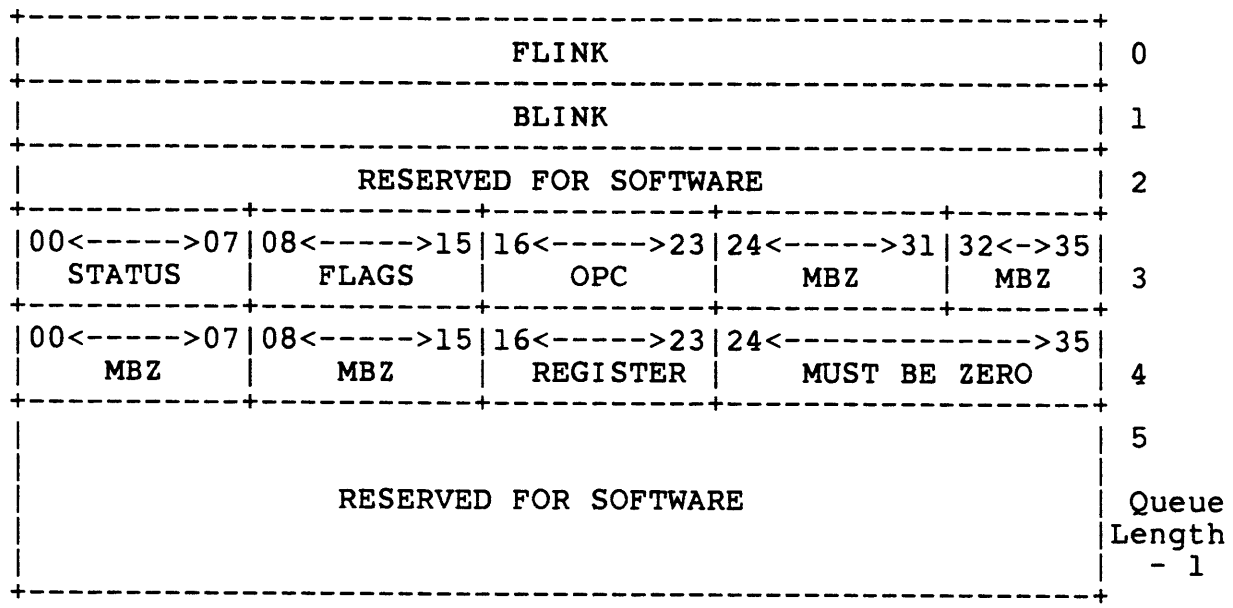
The following are the packet buffer board and link board register definitions for the CI20.

NUMBER	PLI FUNCTION
1	Read Receiver Status
2	Reset Xmit Status
3	Read Buffer
4	Enable Link
5	Read Switches
6	Select Buffer
7	Load Xmit Buffer
10	Read Xmit Status
11	Abort Transmitter
12	Set Mode
13	Transmit
14	Read Node Address
15	Disable Link

16	Release Receiver Buffer
17	Sync

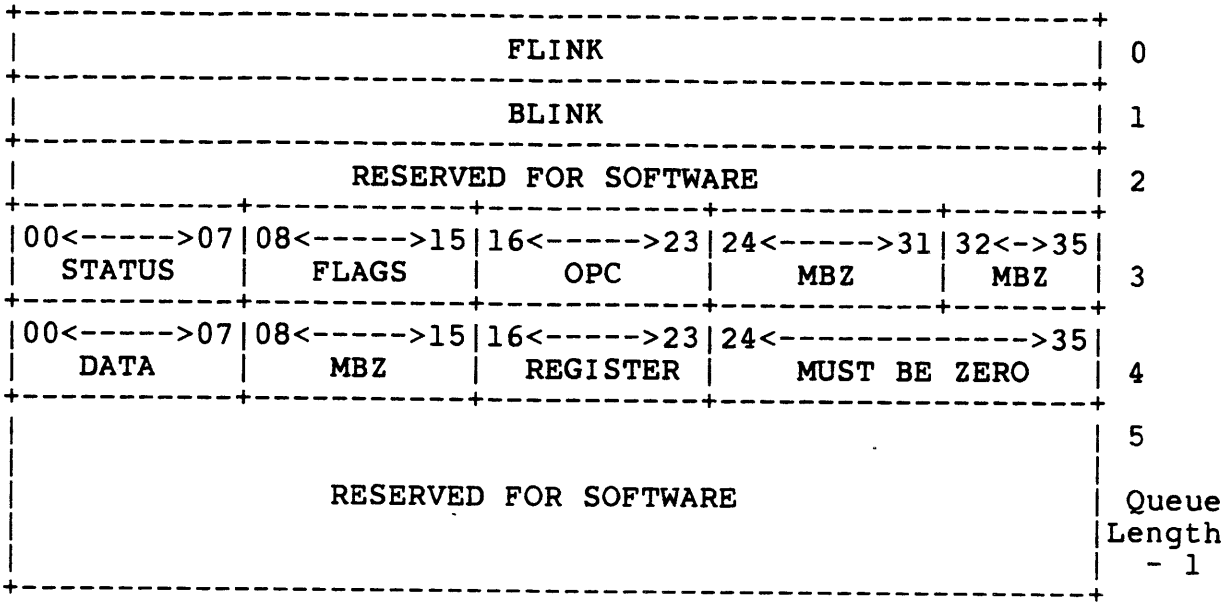
These registers are accessable in all port states except the uninitialized state. For details as to the format of these registers, refer to the PILA Hardware Specification by Shu-Shia Chow Dated 10-Aug-82.

The format of the RDREG command is:



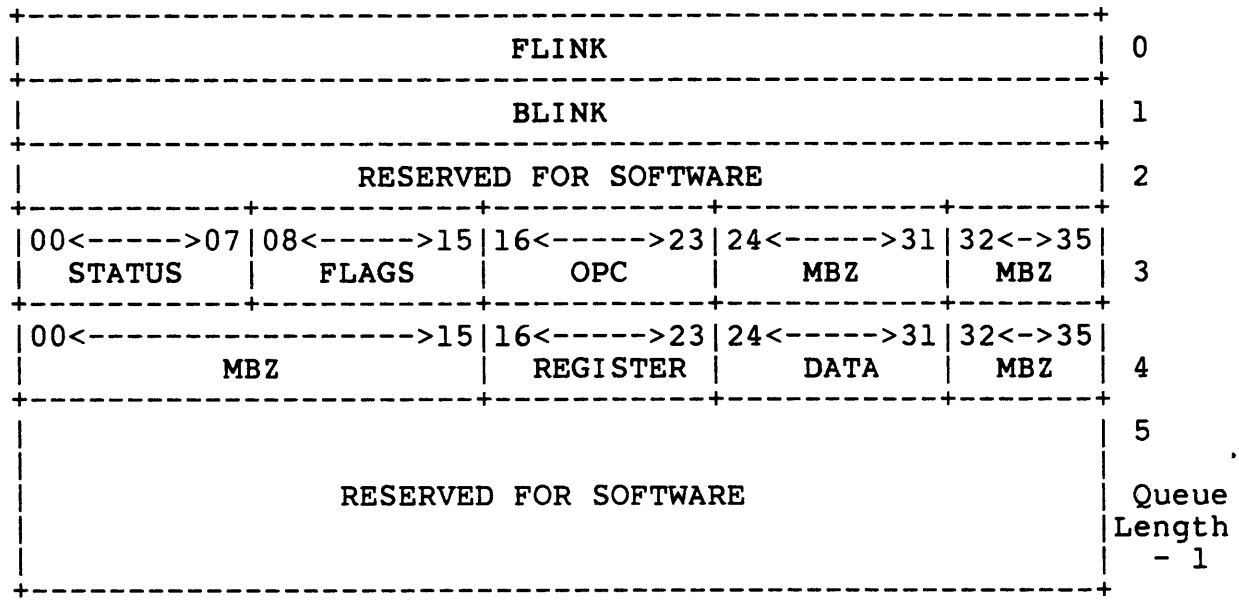
WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 203 octal (RDREG)
4:16-23	REGISTER	This is the register to read.

The format of the Register Read Response (REGRD) follows:



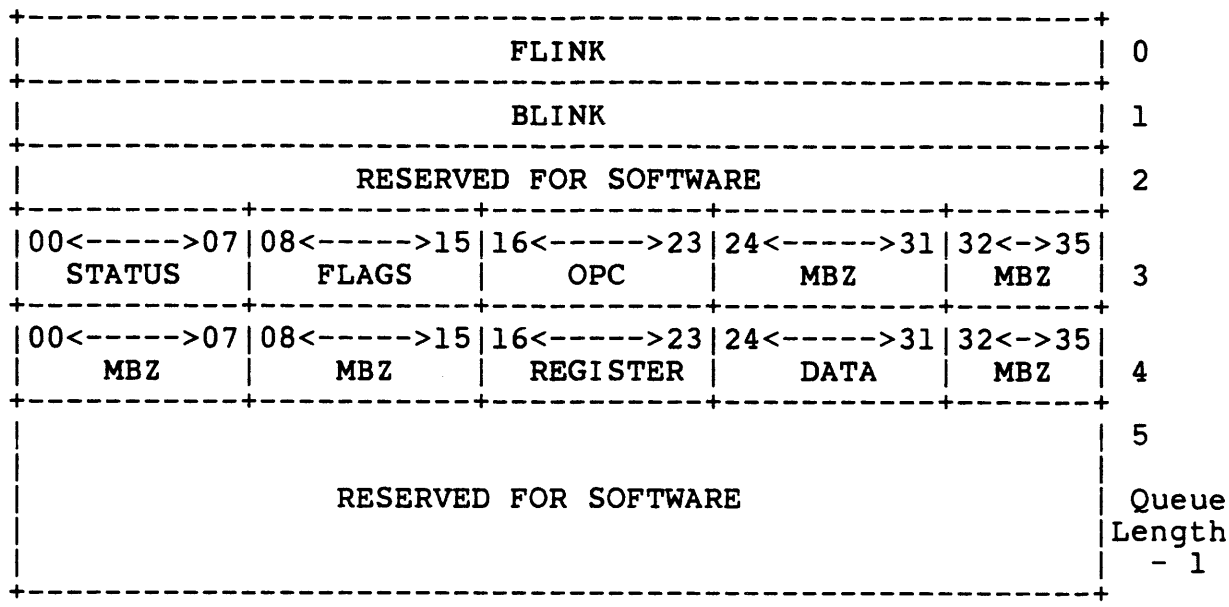
WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 203 octal (REGRD).
4:0-7	DATA	The data from the specified register is returned in this field.
4:16-23	REGISTER	This is the register that was read.

To write a packet buffer board or link register, the Write Register Command (WRREG) is used. The format of this command is:



WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 204 octal (WRREG)
4:16-23	REGISTER	This is the register to write.
4:24-31	DATA	The data from this field is written to the register specified in the Register field.

The format of the Register Written Response (REGWR) is:



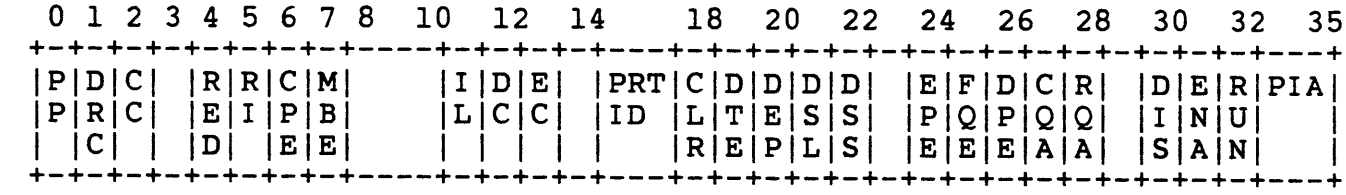
WORD:BITS	NAME	DESCRIPTION
3:16-23	OPCODE	OPCODE = 204 octal (REGWR)
4:16-23	REGISTER	This is the register that was written.
4:24-31	DATA	The data from this field was written to the register specified in the Register field.

14.0 PROGRAMMING NOTES

14.1 PORT/PORT DRIVER COMMUNICATION

The port driver uses KL10 I/O instructions to communicate with and control the port. Starting, stopping and communicating with the port is done by the CONI (CONditions In) and CONO (CONditions Out) instructions. Loading and dumping of the microcode and manipulation of the port while it is not running is done by the DATAI (DATA In) and DATAO (DATA Out) instructions.

The CSR (Control and Status Register) is the mechanism used by the port and the port driver to indicate happenings or events. The format of the CSR is:



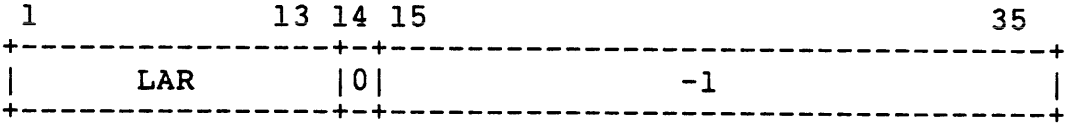
BITS	NAME	DESCRIPTION
0	PORT_PRESENT	This bit is a hardware wire which indicates the presence of a CI20.
1	DIAG_RQST_CSR	Diagnostic purposes only.
2	DIAG_CSR_CHNG	Diagnostic purposes only.
4	RQST_EXAM_DEP	Diagnostic purposes only.
5	RQST_INT	Diagnostic purposes only.
6	CRAM_PARITY_ERROR	Set by the port hardware when a word with bad parity is read from the CRAM.
7	MBUS_ERROR	Set by the port hardware when more than 1 driver connected to the MBUS is detected as being active.
11	IDLE_LOOP	Indicates when the microcode is in the IDLE LOOP.

12	DISABLE_COMPLETE	Indicates the microcode has placed the port in the Disabled state.
13	ENABLE_COMPLETE	Indicates the microcode has placed the port in the Enabled state.
15-17	PORT_ID	A code for the type of port present. Only 1 code is implemented at this time (7).
18	CLEAR_PORT	When set by the port driver the port is placed in a RESET state.
19	DIAG_TEST_EBUF	Diagnostic purposes only.
20	DIAG_GEN_EBUS_PE	Diagnostic purposes only.
21	DIAG_SELECT_LAR	Diagnostic purposes. If this bit is set by the port drivr when the port is not running a DATAI will get the contents of the Last Address Register. The LAR contains the address of the last microinstruction executed.
22	DIAG_SINGLE_CYCLE	Diagnostic purposes only.
24	EBUS_PARITY_ERROR	Set by the hardware when an EBUS Parity Error is detected by the port during an examine operation.
25	FREE_QUEUE_ERROR	Set by the port microcode when it detects that the Datagrm or Message Free Queue in the PCB is empty.
26	DATA_PATH_ERROR	Set by the port microcode when it detects the MVR/FMTR Error Threshold has been reached.
27	CMD_QUEUE_AVAIL	Set by the port driver to indicate that a command has been placed on a previously empty Command Queue in the PCB.

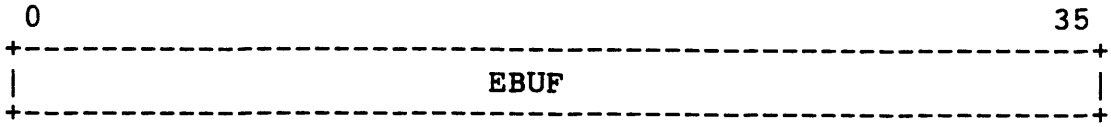
28	RSP_QUEUE_AVAIL	Set by the port microcode when a command has been placed on the Response Queue in the PCB and it was empty.
30	DISABLE	Set by the port driver to put the port into the Disabled state.
31	ENABLE	Set by the port driver to put the port into the Enabled state.
32	MPROC_RUN	Set by the port driver to start the port microcode running.
33-35	PIA	These bits are set by the port driver to indicate what priority level an interrupt by the port will be given.

14.2 LOADING AND DUMPING OF THE PORT

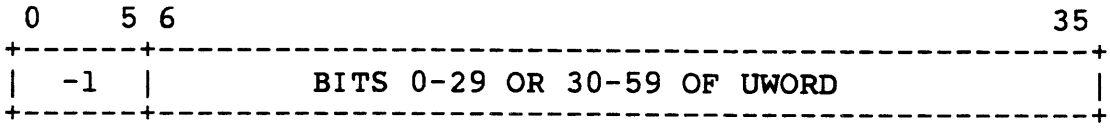
The port microcode generates an interrupt when it sets certain bits in the CSR, A CONI instruction is used by the port driver to read the contents of the CSR. The CONO instruction is used by the port driver to write bits 18-35 of the CSR. When the port driver writes the CSR, bit 2 of the CSR gets set by the port hardware and is sensed by the port microcode. When the port microcode reads the CSR bit 2 in the CSR is cleared. Both DATAI and DATAO instructions have multiple functions with this port. A DATAI with bit 21 of the CSR set and bit 19 of the CSR not set reads the LAR (Last Address Register) in the port. The LAR contains the address of the last micro instruction fetched by the port. The format of the LAR as it is returned by the DATAI is:



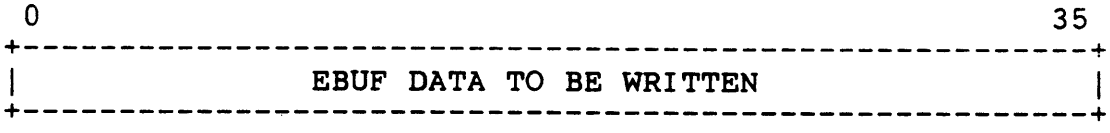
If bit 19 of the CSR is set a DATAI reads the EBUF. The EBUF is the register in the port that is used to communicate over the EBUS. The format of the EBUF is:



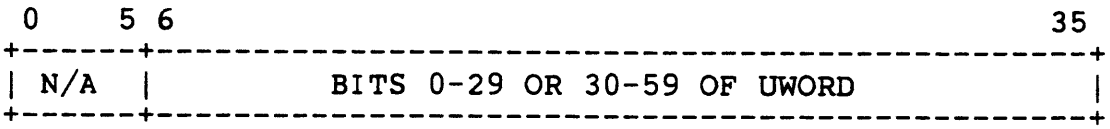
If both bits 19 and 21 of the CSR are not set a DATAI reads either the right or left half of a microword (see DATAO). The format of the microword half returned by the DATAI is:



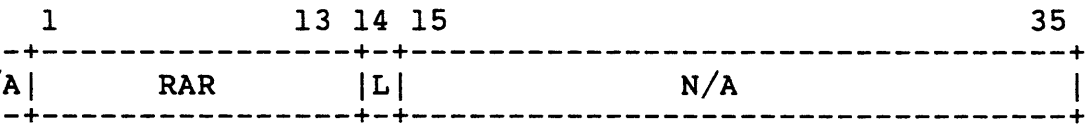
A DATAO with bit 19 of the CSR set will write data to the EBUF. The format of the written data is:



A DATAO with bit 19 of the CSR not set will write data to the right or left half of the microword addressed by the RAR (Ram Address Register). The format of the data to be written is:



A DATAO with bit 0 of the CSR set and bit 19 of the CSR not set will be write the RAR. The RAR is used to address the CRAM for purpose of writing and reading the CRAM. The format of the data to be written is:



If bit 14 is set the operation is performed on the left half of the microword. If bit 14 is not set the operation is performed on the right half.

14.3 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 can be executed. This will guarantee that the port will enter a well-defined state after the reset. It is important to realize that all of the port registers will be set to the power-up state.

14.4 Initialization

When the port is powered-up, there is no valid microcode in the CRAM; valid microcode must be loaded and started. During the initialization, the port microcode will perform a self-check test and generate a planned CRAM Parity Error if it detects any error (see KLCI ERROR SPEC). The microcode will also clear all of the registers.

15.0 ERROR CONDITIONS

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 (see KLCI ERROR SPEC for definition of errors).

While the port is waiting for port driver intervention, it will not be emptying packets out of the receive buffer so packets may be NAKed and Virtual Circuits closed by other ports during this time. The class of non-recoverable errors that cause the port to shut down in this manner are:

1. Inability to get the queue interlock,

2. Non-recoverable internal port error (other than CRAM parity error),

3. Non-recoverable EBUS/CBUS error.

16.0 IMPLEMENTED FUNCTIONALITY

The LCG CI ports will support the following packet formats:

1. DG - send/receive
2. MSG - send/receive
3. SNDDAT - send/receive
4. REQDAT - send/receive
5. RETDAT - send/receive
6. REQID/ID - send/receive
7. CNF - send/receive
8. LB - send/receive
9. REQMDAT - send
10. RETMDAT - receive
11. SNDMDAT - send
12. MCNF - send
13. RST - send
14. STRT - send

If the port is in the enabled state, the port will place any maintenance packet it receives over the CI, except RETMDAT, on the port driver's response queue.

17.0 OPCODE SUMMARY

The following is a table of command and local-response opcode assignments.

COMMAND/RESPONSE	CI PACKET	DEC	OCTAL	BINARY
SNDDG/DGSNT	DG	1	1	0000 0001
SNDMSG/MSGSNT	MSG	2	2	0000 0010
<none>/CNFRET	CNF	3	3	0000 0011
REQID/IDREQ	IDREQ	5	5	0000 0101
SNDRST/RSTSNT	RST	6	6	0000 0110
SNDSTRT/STRTSNT	STRT	7	7	0000 0111
REQDAT0/DATREQ0	DATREQ0	8	10	0000 1000
REQDAT1/DATREQ1	DATREQ1	9	11	0000 1001
REQDAT2/DATREQ2	DATREQ2	10	12	0000 1010
SNDLB/LBSNT	LB	13	15	0000 1101
REQMDAT/MDATREQ	MDATREQ	14	16	0000 1110
SNDDAT/DATSNT	SNTDAT	16	20	0001 0000
<none>/DATRET	RETDAT	17	21	0001 0001
SNDMDAT/MDATSNT	SNTMDAT	18	22	0001 0010
SETCKT/CKTSET	<none>	128	200	1000 0000
SETCNT/CNTSET	<none>	129	201	1000 0001
RDCNT/CNTRD	<none>	130	202	1000 0010
RDREG/REGRD	<none>	131	203	1000 0011
WRTREG/REGWRT	<none>	132	204	1000 0100
CLSBUF/BUFCLS	<none>	133	205	1000 0101

The following table contains the opcode assignments for remotely-generated responses.

RESPONSE	CI PACKET	DEC	OCTAL	BINARY
DGREC	DG	33	41	0010 0001
MSGREC	MSG	34	42	0010 0010
CNFREC	CNF	35	43	0010 0011
MCNFREC	MCNF	36	44	0010 1001
IDREC	ID	43	53	0010 1011
LBREC	LB	45	55	0010 1101
DATREC	RETDAT	49	61	0011 0001
MDATREC	RETMDAT	51	63	0011 0011

INDEX