19.0 IP APPLICATION INTERFACE

19.1 IP Interface Overview

The IP AI is provided to application programs written in FORTRAN or any language that can interface to FORTRAN (for example, SYMPL and COBOL).

19.2 Common IP Interface Parameters

There are some parameters, large arrays with several entries, which are required in several different IP calling sequences. These are:

1) IP Source Address Parameter
2) IP Destination Address Parameter
3) IP Header Record
4) IP Options Record
5) ICMP Header Record
6) IP Return Status
7) Wait for Reply

For convenience and consistency, these parameters are described in detail in the following subsections, and then are briefly described in each routine that uses the parameter.

19.2.1. IP Source Address (SOURCE_ADDR)

The IP source address is a 3-word array that specifies the IP source address of the datagram. This is the address of your application host.

Word	Description
1	Address Fields In-use. Indicates which IP address fields are specified and which are unspecified. In some circumstances, it is possible to leave fields unspecified

<u>Word</u> <u>Description</u>

in which case IP will assign values to the fields. The following in-use codes are defined:

In-use = 0: Neither the host nor the network address is specified. Both words 2 and 3 of the array are ignored. If neither host nor network is specified, then IP will assign the default network and host address of the CDCNET Device Interface where the IP gateway resides for this host. If the DI has more than one source address, the default is chosen based upon the destination network.

In-use = 1: The network address is specified, but the host address is not specified. Word 3 of the array is ignored. If the host is unspecified, then IP will assign the default host address of the CDCNET Device Interface for the specified network. The DI must have an address on the specified network or the address is incorrect.

In-use = 2: The host address is specified, but the network address is not specified. Word 2 of the array is ignored. If the network is unspecified, then IP will assign the network address of the CDCNET Device Interface. If the DI has more than one network address then the network will be chosen based upon the destination address. The DI must have the specified host address for the chosen network or the address is incorrect.

In-use = 3: Both the host and network addresses are specified. The DI must have the specified host address on the specified network or the address is incorrect.

2	Network Address. Specifies the network portion of the source internet address. This field is meaningful only if In-use is set to 1 or 3. The portion of the 32-bit internet address that is the network address depends upon the network class. It can take values in the range 0..0FFFFFF(16). Refer to the general discussion in the subsection above about internet addresses.
3	Host Address. Specifies the host portion of the source internet address. This field is meaningful only if In-use is set to 2 or 3. The portion of the 32-bit internet address that is the host address depends upon the network class. It can take values in the range 0..0FFFFFF(16). Refer to the general discussion in the subsection above about internet addresses.

19.2.2 IP Destination Address (DEST_ADDR)

The IP destination address is a 3-word array that specifies the IP destination address of the datagram. This is the address of the host that you wish to exchange data with.

<u>Word</u>	<u>Description</u>
1	Address Fields In-use. Indicates which IP address fields are specified and which are unspecified. The IP interface does not permit either an unspecified host or the network destination address. The following in-use codes are defined: In-use = 0: Neither the host nor the network address is specified. An error is returned if this value is used with the IP interface.

Word	Description
	In-use = 1: The network address is specified, but the host address is not specified. An error is returned if this value is used with the IP interface.
	In-use = 2: The host address is specified, but the network address is not specified. An error is returned if this value is used with the IP interface.
	In-use = 3: Both the host and network addresses are specified. This is the required type for IP interface requests.
2	Network Address. Specifies the network portion of the destination internet address. This field is meaningful only if In-use is set to 1 or 3. The portion of the 32-bit internet address that is the network address depends upon the network class. It can take values in the range 0..OFFFFFF(16). Refer to the general discussion in the subsection above about internet addresses.
3	Host Address. Specifies the host portion of the destination internet address. This field is meaningful only if In-use is set to 2 or 3. The portion of the 32-bit internet address that is the host address depends upon the network class. It can take values in the range 0..OFFFFFF(16). Refer to the general discussion in the subsection above about internet addresses.

19.2.3 IP Header Record

The IP Header Record allows an application to specify some of the fields in the IP header of an IP datagram, and to receive these values for datagrams received from a peer application.

If you do not wish to specify IP header fields, which is usually the case, then use a value of zero for an input parameter, and it is not necessary to allocate an array for the record. When an IP header record appears as an output parameter, or when you wish to specify any IP header fields in an input parameter, then the parameter must be a 21-word word array. Some of the fields are output-only; if you specify these fields on in an input record, they will be ignored.

Although some description of the IP header fields is provided, you must have some knowledge of the IP protocols to use most of the fields correctly. Refer to MIL-STD 1777 for more information. In general, you do not need to know or understand the IP header fields, and you can use the default values.

A list of the words in the array and their purpose is given below, followed by a more detailed description of each word.

Word	Summary
1	IP Header In-use, in/out, 0 or -1.
2	Full Header In-use, in/out, 0 or -1.
3	IP Version, out, 0..15.
4	IP Header Length, out, 0..15.
5	Precedence, in/out, 0..7.
6	Delay, in/out, 0 or -1.
7	Throughput, in/out, 0 or -1.
8	Reliability, in/out, 0 or -1.
9	Reserved.

Word	Summary
10	Reserved.
11	Total IP Packet Length, out, 0..0FFFF(16).
12	Packet Identification, in/out, 0..0FFFF(16).
13	Reserved.
14	Don't Fragment, in/out, 0 or -1.
15	Fragmented, out, 0 or -1.
16	Fragment Offset, Unused.
17	Time-to-Live, in/out, 0..256.
18	IP Protocol, in/out, 0..256.
19	Checksum, Unused.
20	Source Address, out, 32 bits.
21	Destination Address, out, 32 bits.

Word	Description
1	IP Header In-use. If 0 then the remainder of the record is ignored (and does not need to be allocated) on input. If -1, then the fields in use are identified by the value of word 2. If you do not need to specify any header values, this should be set to 0 to reduce the size of packets sent between the CYBER and the IP gateway (approximately 20 words).
2	Full Header In-use. If 0 then only the protocol and packet identifier fields are in use, and all other fields are ignored. If -1, then all fields in the header record are in use. If you do not need to specify any header values other than protocol and packet identifier, this should be set to 0 to reduce the size of packets sent between the CYBER and the IP gateway (approximately 20 words).

3 IP Version. This is an output-only parameter and is ignored on input. It contains the version of the Internet Protocol currently in use, and is a value in the range 0..15. The current value is 4.

4 IP Header Length. This is an output-only parameter and is ignored on input. It contains the number of bytes in the packed IP header, including any IP options.

5 Precedence. Specifies the precedence of the datagram, and can take values of 0..7. The meanings of the values are as follows:

0 - Routine
1 - Priority
2 - Immediate
3 - Flash
4 - Flash Override
5 - CRITICAL/EGP
6 - Internetwork Control
7 - Network Control

Most networks such as ARPANET restrict the range of precedence for a particular host or network to a certain range; usually only 0 is permitted. Your site administrator may also restrict the precedence of your network link. When no IP header record is specified, the default precedence is 0.

6 Delay. Indicates whether delay is an important consideration when choosing a route or resources for the datagram. The delay in an IP transmission is the elapsed

time that it takes an IP packet to reach its destination. The choices made by IP in a transmission involve tradeoffs between delay, throughput (see word 7), and reliability (see word 8). You should specify -1 for Delay when it is relatively important for your application, or 0 if Delay is not important. For example, Delay is not important in file transfer applications, but it is important in network voice applications. When no IP header record is specified, the default Delay is 0.

7 Throughput. Indicates whether throughput is an important consideration when choosing a route or resources for the datagram. The throughput in an IP transmission is the average number of bytes of data that can be transmitted in a fixed period of time. The choices made by IP in a transmission involve tradeoffs between delay (see word 6), throughput, and reliability (see word 8). You should specify -1 for throughput when it is relatively important for your application, or 0 if throughput is not important. For example, throughput is not important in typical interactive applications, but is important in file transfer applications. When no IP header record is specified, the default throughput is 0.

8 Reliability. Indicates whether reliability is an important consideration when choosing a route or resources for the datagram. The reliability in an IP transmission is the average number of bytes of data lost or corrupted per fixed number of bytes. The choices made by IP in a transmission involve tradeoffs between delay (see word 6), throughput (see word 7), and reliability. You should specify -1 for reliability when it is relatively important for your application, or 0 if

reliability is not important. When no IP header record is specified, the default reliability is 0.

9 Reserved.

10 Reserved.

11 Total IP Packet Length. This is an output-only parameter and is ignored on input. It is the total length of the IP datagram.

12 Packet Identification. This specifies an indentifier that your application can use to distinguish one datagram from another. If the header is omitted, the IP gateway will start with 0 for the first datagram and increment the value for each datagram sent.

13 Reserved.

14 Don't Fragment. This specifies whether IP can fragment the datagram. Different networks have different limits on the maximum IP data packet size, and an IP packet that was less than the maximum at the source network may reach a network where the same packet is larger than the maximum. If Don't Fragment is set to 0, then IP will break the packet into small enough packets for the network, and then IP at the destination will re-assemble the packet. If Don't Fragment is set to -1, then IP will reject the packet and an error indication (IPEI) will be delivered. When no IP header record is specified, the default Don't Fragment is 0.

15 Fragmented. This is an output-only parameter and is ignored on input. It is 0 if the datagram was not fragmented, or -1 if it was fragmented. IP always re-assembles a datagram that was fragmented before it delivers the datagram to your application.

16 Fragment Offset. This field has no meaning for an application and is ignored.

17 Time-to-Live. This specifies the IP time-to-live (TTL) for the datagram. TTL is usually treated as a hop count, where each IP that forwards a packet decrements the count; sometimes it is also decremented on the basis of time (in seconds) within the IP transmission queue. When the TTL reaches zero, IP discards the packet and sends an error back to your host, which results in an error indication (IPEI).

18 IP Protocol. The IP protocol field identifies the protocol of the application that is to receive the datagram. Your application must open a SAP for each protocol for which it wishes to receive datagrams. A datagram can be sent however, to any protocol regardless of the protocol specified in the SAP request. Protocol numbers are assigned by SRI, and are listed in RFC-790 ASSIGNED NUMBERS.

19 Checksum. This field has no meaning for the IP interface and is ignored.

20 Source Address. This is an output-only parameter and is ignored on input. It is the 32-bit packed IP source address for the datagram, right-justified in a 60-bit word.

21 Destination Address. This is an output-only parameter and is ignored on input. It is the 32-bit packed IP destination address for the datagram, right-justified in a 60-bit word.

19.2.4 IP Options Record

The IP Options Record allows an application to specify the IP options in the header of datagram.

If you do not wish to specify IP header options, which is usually the case, then use a value of zero for an input parameter, and it is not necessary to allocate an array for the record. When an IP option record appears as an output parameter, or when you wish to specify any IP header options in an input parameter, then the parameter must be a 39-word word array.

Although some description of the IP options is provided, you must have some knowledge of the IP protocol to use most of the options correctly. Refer to MIL-STD 1777 for more information. In general, you do not need to know or understand the IP options, and you can use the default values.

A list of the words in the array and their purpose is given below, followed by a more detailed description of each word.

Word	Option	Summary
1	IP Options	In-use, in/out, 0 or -1.
2	Reserved.	
3	Security	In-use, in/out, 0 or -1.
4		Level, in/out, 0..0FFFF(16).
5		Compartments, in/out, 0..0FFFF(16).
6		Handling Restrictions, in/out, 0..0FFFF(16).
7		Transmission Control Code, in/out, 0..0FFFFFF(16).

Word	Option	Summary
8	Reserved.	
9	Stream-Id	In-use, in/out, 0 or -1.
10		Stream Identifier, in/out, 0..FFFF(16).
11	Reserved.	
12	Routing	In-use, in/out, 0 or -1.
13		Type of Routing, in/out, 0..2.
14		Next in List, out, 0..255.
15		Length of List, in/out, 0..255.
16-24		List of IP Addresses, in/out, each 32 bits.
25	Reserved.	
26	Timestamp	In-use, in/out, 0 or -1.
27		Type, in/out, 0..3.
28		Next in List, out, 0..255.
29		Length of List, in/out, 0..255.
30		Overflow Count, out, 0..127.
31-39		List of Timestamps, in/out, each 32 bits.

Word	Description
1	IP OptionsIn-use. If 0, then the remainder of the record is ignored (and does not need to be allocated) on input. If -1, then the options specified depend upon the setting of the in-use flag for each option.
2	Reserved.
3	Security Option In-use. If 0, then words 4-7 have no meaning and should be ignored. If -1, then the fields for the security option are given in words 4-7. This option provides a way to send security information through the network in a standard manner.

4 Security Option Level. Specifies the security level of the datagram, and is a 16-bit field, right-justified in a 60-bit word. The NOS and CDCNET implementation of security is strictly system-high. That is, all elements in the system from the secured network connection to the CYBER must all operate at the same security level if true guaranteed security is required. A site can force the security to a certain level, in which case any attempt to specify a different level will cause the datagram to be discarded. If the site does not enforce a particular level, you can specify any level but this level of security is not guaranteed. The following values are defined for the security level:

0	- Unclassified
61749 or 0F135(16)	- Confidential
30874 or 789A(16)	- EFTO
48205 or 0BC4D(16)	- MMMM
24102 or 05E26(16)	- PROG
44819 or 0AF13(16)	- Restricted
55210 or 0D7AA(16)	- Secret
27589 or 6BC5(16)	- Top Secret
13794 or 35E2(16)	- Reserved
39665 or 9AF1(16)	- Reserved
19832 or 4D78(16)	- Reserved
9405 or 24BD(16)	- Reserved
4958 or 135E(16)	- Reserved
35247 or 89AF(16)	- Reserved
50390 or 0C4D6(16)	- Reserved
57963 or 0E26B(16)	- Reserved

5 Security Option Compartments. Specifies the security compartments for the connection. It is a 16-bit field, right-justified in a 60-bit word. If zero, then no compartments are specified. The definitions of compartments can be obtained from DCA.

6 Security Option Handling Restrictions. Specifies the control and release markings for the connection. If zero, then no handling restrictions are specified. The values are alphanumeric digraphs which are defined in the DIA manual DIAM 65-19, "Standard Security Markings".

7 Security Option Transmission Control Code. The TCC provides a means to segregate traffic and define controlled communities of interest among subscribers. If zero, then no TCC values are specified. The values are trigraphs and the definitions can be obtained from DCA.

8 Reserved.

9 Stream Option In-use. If 0, then word 10 has no meaning and should be ignored. If -1, then the stream identifier is given in word 10. This option provides a way to send SATNET stream identifiers through networks that do not support the stream identifier.

10 Stream Option Identifier. Specifies the 16-bit stream identifer for the packet, right-justified in a 60-bit word.

11 Reserved.

12 Routing Option In-use. If 0, then words 13-24 have no meaning and should be ignored. If -1, then the fields for the routing option are given in words 13-24. This option provides a means to record and control the route of datagrams.

13 Routing Option, Type of Routing. Specifies the variant of routing option that is in use. The following types are permitted:

Type = 0: Strict source and record routing option. Allows the application to specify a particular route for datagrams, and to specify that the route be recorded as IP routes the packet from one network to another. Normally, IP chooses its own route for packets; this option forces a specific route to be taken. If the requested route cannot be honored, IP discards the packet and returns an error indication (IPEI). The requested route should be stored as a list of IP addresses in words 15-24.

Type = 1: Loose source and record routing option. Allows the application to specify a particular route for datagrams and to specify that the route be recorded as IP routes the packet from one network to another. Normally, IP chooses its own route for packets; this option forces a specific route to be taken. This option differs from strict routing in that IP may choose a different route if it cannot honor the requested route, and the packet will not be discarded. The requested route is stored as a list of IP addresses in words 15-24 when issuing a datagram request.

Type = 2: Record route option. Allows the application to specify that the route be recorded as IP routes datagrams from on network to another.

Route control and recording specified in a datagram occurs only in the direction of local-to-peer. If you need control and recording of datagrams sent by the destination application to your application, then the destination application must request the IP routing options.

14 Routing Option, Next in List. An integer in the range 0..255 that is an index into the array consisting of words 15-24 (where index=0 refers to word 15). The index specifies the next free entry in the array. Although the array can handle a maximum of 9 entries, the index may be greater. In this case, the list has overflowed and the most recent entries are lost. The Next parameter is an output-only parameter and is ignored on input.

15 Routing Option, Length of List. An integer in the range 0..255 that specifies the total number of IP address entries in the array consisting of words 15-24. Although the array can handle a maximum of 9 entries, the length may be greater. In this case, the list has overflowed and the most recent entries are lost.

16-24 Routing Option, List of IP Addresses. This is a 9-word array of 32-bit packed IP addresses, each right-justified in a 60-bit word. The number of entries in use is determined by word 15. On input, this array should contain the IP addresses of gateways that should be used to reach the destination. On output, this array contains the IP addresses of gateways that were used to send datagrams from the destination, but only if the destination application selected a routing option.

25 Reserved.

26 Timestamp Option, In-use. If 0, then words 27-39 have no meaning and should be ignored. If -1, then the fields for the timestamp option are given in words 27-39. This option provides a means to record timing and routing information as a datagram is routed between networks.

27 Timestamp Option, Type of Timestamp. Specifies the variant of timestamp option that is in use. The following types are permitted:

Type = 0: Record Timestamps. As a datagram is routed from one network to another, the gateway records the current time in a timestamp list.

Type = 1: Record Timestamps and IP Addresses. As a datagram is routed from one network to another, the gateway records both the current time and the internet address of the gateway in a timestamp list.

Type = 2: Reserved.

Type = 3: Selectively Record Timestamps. Allows the application to specify a list of IP addresses of gateways that are to record timestamps. Only the timestamps of the specified gateways in the route will be recorded; any other gateways will route the datagram without recording a timestamp. The IP addresses are specified in words 31-39 when issuing a datagram request.

Timestamp recording specified for a datagram occurs only in the direction from the local application to the peer. If you need recording of datagrams sent by the peer to your application, then the peer application must request the IP timestamp options.

28 Timestamp Option, Next in List. An integer in the range 0..255 that is an index into the array consisting of words 31-39 (where index=0 refers to word 31). The index specifies the next free entry in the array. Although the array can handle a maximum of 9 entries, the index may be greater. In this case, the list has overflowed and the most recent entries are lost. The Next parameter is an output-only parameter and is ignored on input.

29 Timestamp Option, Length of List. An integer in the range 0..255 that specifies the total number of timestamp entries in the array consisting of words 31-39. Although the array can handle a maximum of 9 entries, the length may be greater. In this case, the list has overflowed and the most recent entries are lost.

30 Timestamp Option, Overflow Count. An integer in the range 0..127 that specifies the number of IP gateways that were unable to record timestamp information because the IP option was full. Note that this is different from the overflow described in the length field (word 29), where the entries were recorded but could not fit within the fixed 9-word array. The Overflow parameter is an output-only parameter and is ignored on input.

31-39 Timestamp Option, List of Timestamps. This is a 9-word array of 32-bit timestamps, or of pairs consisting of a 32-bit timestamp and a 32-bit packed IP address, with each 32-bit value right-justified in a 60-bit word. The format of the array, either timestamps or timestamp/address pairs, is determined by the timestamp type (word 27).

If the high-order bit of the timestamp (bit 31 using COMPASS numbering) is clear, then the timestamp represents the number of milliseconds since midnight, universal time (UT). If the bit is set, then the time may be some other format that is specific to the IP gateway site.

On input, if selective recording is specified, then this each timestamp/address pair should contain the IP addresses of the gateways that will record timestamp information. On output, this array contains the timestamps, and optionally IP addresses, of gateways that were used to send datagrams from the destination peer, but only if the destination peer selected a timestamp option.

19.2.5. ICMP Header Record

The ICMP Header Record allows an application to specify some of the fields of an ICMP message, and to receive these values for ICMP messages received from a peer application. It is a 4-word array, where the contents of the fourth word is determined by the ICMP message type in word 1.

Although some description of the ICMP header fields is provided, you must have some knowledge of the IP protocols to use most of the fields correctly. Refer to MIL-STD 1777 and RFC-792 for more information.

A list of the words in the array and their purpose is given below, followed by a more detailed description of each word.

Word	Summary
1	ICMP Message Type, in/out, 0..255.
2	Detailed Code, in/out, 0..255.
3	ICMP Checksum, out, 0..0FFFF(16).
4a	Gateway Address, in/out, 32 bits.
4b	Pointer to Erroneous Value, in/out, see below.
4c	Echo Identifier and Sequence, in/out, see below.

<u>Word</u> <u>Description</u>

1 ICMP Message Type. Specifies the ICMP message type. The common message types are:

(0) Echo Reply. The message is a reply to a previous echo request. Word 4 of the record contains the identifier and sequence number supplied by your application in the echo request. Any data supplied in the request is returned in the reply. IP applications should not normally issue this message, since IP will automatically reply to any received echo requests.

(3) Destination Unreachable. Indicates that the specified destination for a datagram could not be reached. The reason is given in word 2, which can take on the following values:

0 Network Unreachable.
1 Host Unreachable.
2 Protocol Unreachable.
3 Port Unreachable.
4 Fragmentation Needed and Don't Fragment Set
5 Source Route Failed.

An application will normally issue this message with code 3 only, since IP automatically issues unreachable messages for codes 0-2, 4, and 5.

The data associated with this message contains the IP header of the original datagram, followed by the first 64-bits of the data from the datagram.

(4) Source Quench. Indicates that the source of the ICMP message is congested, and has either started discarding datagrams or will soon be forced to do so. The receiver of this message should wait some period of time before sending data to this destination. An IP application can send this message if it cannot handle the volume of datagrams it is receiving from its peers.

The data associated with this message contains the IP header of the original datagram, followed by the first 64-bits of the data from the datagram.

(5) Redirect. Indicates that a previously sent datagram was sent to the wrong gateway in a route to the destination host. The gateway sends this ICMP message to indicate the correct gateway address, which is in word 4 of the header record. Since applications that use the IP interface do not normally choose the route of a datagram, this is usually informative and does not require any action by the application. IP will automatically change the route used for subsequent datagrams to the specified destination. IP also automatically issues redirects when it receives misdirected packets, so that applications should not normally issue redirect messages.

Word 4 of the record contains the internet address of the new gateway to use. Word 2 of the record contains the type of redirect. The following codes may be sent or received:

0 Redirect datagrams for the Network
1 Redirect datagrams for the Host
3 Redirect datagrams for the Type of Service and the Network
4 Redirect datagrams for the Type of Service and the Host. The data associated with this message contains the IP header of the original datagram, followed by the first 64-bits of the data from the datagram.

(8) Echo Request. When received, indicates that the source is requesting an echo reply, containing the data, identifier, and sequence number. For applications, this is informative, since IP will automatically reply to any echo request. Applications can issue an echo requests to other hosts as a debugging aid, for example, to determine host reachability.

(11) Time Exceeded. Indicates that the time-to-live field (TTL) of a datagram reached zero, and the datagram was discarded. This can happen when a) the datagram was forced to traverse more gateways, networks, or intermediate systems than specified in the TTL, b) the datagram spent more time in seconds queued in gateways in the network than specified in the TTL, d) all the fragments of a fragmented datagram were not received within the number of seconds specified in the TTL. The code given in word 2 is set to 0 if the TTL was exceeded in transit and is set to 1 if the fragment reassembly

time was exceeded. IP automatically issues time exceeded messages if the TTL reaches zero, so an IP application should not normally send this message. The data associated with this message contains the IP header of the original datagram, followed by the first 64-bits of the data from the datagram.

(12) Parameter Problem. Indicates that some problem in the IP header prevented a gateway or host from processing the datagram, and so the datagram was discarded. Word 4 contains a pointer to the byte number of the incorrect IP header entry. IP automatically sends this message when it discards a datagram because of some header problem, so an IP application should not normally send this message. The data associated with this message contains the IP header of the original datagram, followed by the first 64-bits of the data from the datagram.

(13) Timestamp. When received, indicates that the source is requesting a timestamp reply, containing the timestamps, identifier, and sequence number. For applications, this is informative, since IP will automatically reply to any timestamp request. Applications can issue timestamp requests to other hosts as a debugging aid, for example, to estimate the round trip time to another host. The format of the timestamps is given in the timestamp reply (message type 14) description below.

(14) Timestamp Reply. The message is a reply to a previous timestamp request. Word 4 of the record contains the identifier and sequence number supplied by your application in the timestamp request. Three 32-bit timestamps are returned in the data associated with the

message. They are packed into 2 60-bit words, with 24 bits of trailing unused space. The first timestamp is the time that the timestamp request was sent, the second is the time that the destination IP received the request, and the third is the time that the destination IP send the reply. All timestamps are in the same format used by the timestamp option of the IP header record, described in the previous subsection. IP applications should not normally issue this message, since IP will automatically reply to any received timestamp requests.

(15) Information Request. When received, indicates that the source is requesting an information reply, containing the IP source and destination addresses, and also the identifier and sequence number. For applications, this is informative, since IP will automatically reply to any echo request. Applications should not normally send this request either.

(16) Information Reply. The message is a reply to a previous information request. Word 4 of the record contains the identifier and sequence number supplied by your application in the information request. The IP header associated with this reply contains the source and destination addresses supplied in the reply. IP applications should not normally issue this message, since IP will automatically reply to any received information requests.

2 Detailed Code. Specifies a message code that is specific to a particular ICMP message type. Refer to the message types above for the meaning of the codes. Many messages do not use this field, in which case it is zero.

3 ICMP Checksum. Specifies the checksum of the ICMP message. This field is output only, and is ignored on input.

4a Gateway Address. This variation of word 4 is used in conjunction with the ICMP redirect message (word 1 = 5). The value specifies the address of the gateway to be used for the given destination address. It is a 32-bit packed IP address, right-justified in a 60-bit word.

4b Pointer to Erroneous Value. This variation of word 4 is used in conjunction with the ICMP parameter problem message (word 1 = 12). The value identifies the byte number within the IP header (numbered from zero) that was incorrect.

4c Echo Identifier and Sequence. This variation of word 4 is used in conjunction with the ICMP echo request, echo reply, timestamp request, timestamp reply, information request, and information reply messages (word 1 = 0, 8, 13, 14, 15, or 16). Bits 15-0 contain the 8-bit message sequence number, and bits 31-16 contain the 16-bit message identifier. The application will typically use the identifier to identify a particular session, while it will use the sequence number to identify a particular echo/timestamp/information request within that session. If the application uses an identifier larger than 8-bits, then IP will truncate the upper bits and replace them with the IP protocol associated with the SAP, so that IP can route the reply to the appropriate application.

19.2.6 IP Return Status (STATUS)

The IP return status variable is an integer variable that is returned by most IP library routines. It reports the status of your IP request. When an IP request has successfully completed, a code of 20 is returned. Any other code means that there was an error and the request could not be completed.

If you call an IP routine with wait=FALSE (refer to the subsection on 'wait' below), then status may be negative. This indicates that the request has been started, but has not yet been completed. You must periodically check the status variable until it becomes positive. You cannot change the value of the status variable, or use it for any other purpose as long as it has a negative value.

Following is a description of the possible return codes, and their meanings.

Code	Description
1	No indications or status returned. This value is returned by an IPAI request when there are no indications available and no incomplete requests (refer to the 'wait' subsection) have completed.
2	Previous operation incomplete. Your application has issued a request that depends upon a previous request that has not completed. For example, you cannot open another IP SAP (IPOS) until the previous open SAP request has completed. Your application must wait until the previous operation completes before it re-issues the request.

3 Resources busy, try again later.

The resources required to perform your request are not available. You must wait a short period of time and then try again. This can occur under the following circumstances:

1. You have issued consecutive requests without intervening calls to IPAI. This causes your NAM application block limit to be reached because the acknowledgments have not been received. You must restructure the application to call IPAI more frequently, or you must retry the request later. This code does not occur if you are using wait=TRUE.

2. You have issued several consecutive requests to IPOS. This routine reserves some space within the runtime routines that is limited, and sometimes all of the space may be reserved. You must wait until one of the requests completes. This code does not occur if you are using wait=TRUE.

4 Application did not call DDNRA first.

The application has not called DDNRA prior to the IP request. This code may also be returned after a forced network shutdown has occurred.

5 Network unavailable; the network link has been lost.

IP requests cannot be honored because the network link is no longer available. This response is returned if the operator causes a forced network shutdown, or when the link to the CDCNET IP Gateway has been abruptly terminated. You must reissue the DDNRA and IP Open SAP (IPOS) requests to regain IP access. Depending upon the cause of the code, this may be several minutes.

6 SAP inactive.

This is returned by an IP error indication (IPEI) when the indicated SAP has been inactive for some site-determined period of time (usually about 10 minutes). This is an informative code, and the application must determine if this inactivity represents an error.

7-8 Reserved.

9 Indication delivered.

This is returned by IPAI if one of the application's IP indication handlers was called. If both a request status and an indication are delivered during the same IPAI call, then code 10 is returned.

10 Request status delivered.

This is returned by IPAI if one of the application's outstanding requests, issued with wait=FALSE, has completed. The application can now examine the reply status variable for the request to determine success or failure. If both a request status and an indication are delivered during the same IPAI call, then a code 10 is returned.

11 The supplied status variable is already in use.

The status variable supplied in the request was already in use for a previous request (with wait=FALSE) that was not yet completed. The IP interface routines use this status variable to track outstanding requests, and IP cannot now report the status of the previous request when the request does complete. Therefore, the SAP associated with the previous request is aborted, and the application is notified using a close SAP indication.

12 Network idledown in progress.

The operator has initiated an idledown, which means that in a minute or two the network will shutdown. This code is returned via an IP error indication for each IP SAP. The application should notify any users of the pending shutdown, close SAPs as soon as possible and issue a DDNDR request.

13-19 Reserved.

20 Successful request.

The IP request has completed and was successful.

21 Protocol already in use.

This is returned in response to an IP Open SAP request (IPOS) when the specified protocol is already in use by another application.

22 Incorrect protocol.

The application has entered an incorrect value for the protocol in an IP Open SAP (IPOS) request. This is returned when the protocol is out of range (greater than 255).

23 SAP does not exist, incorrect IP SAPID.

The SAP specified by the IP SAPID does not exist. The application may have used an incorrect IP SAPID, or the SAP may not have been opened, or a previously opened SAP may have closed or aborted. You should verify that the application is using the IP SAPID returned by an IPOS request, that the SAPID is not used until a successful response to the IPOS request is received, and that the application does not use the SAPID after it has issued a close SAP request (IPCS) or after it has received a close SAP indication (IPCSI) for the SAP.

24 Unreachable network address.

The network at the IP destination address given in the datagram could not be reached. This is returned when the local or intermediate network does not know how to route the datagram to the given network. Either: 1) the address is incorrect, 2) your site does not have the specified network correctly configured, or 3) a gateway which provides access to the destination network is currently not operating. Check the address and correct it, or notify your site administrator. In some networks or hosts, the datagram may simply be discarded without notice. This code can be returned in response to an ICMP destination unreachable message received for your application.

25 Unreachable host address.

The host at the IP destination address given in the datagram could not be reached. This is returned when the destination network does not know how to route the datagram to the given address. Either: 1) the address is incorrect, 2) your site does not have the specified host correctly configured, or 3) the remote site where the host resides does not have the specified host correctly configured. Check the address and correct it, or notify your site administrator. This code is not usually returned if the host exists but is dropping the datagram; instead, the datagram is discarded without notice. It can be returned in response to an ICMP destination unreachable message received for your application.

26 Unreachable protocol.

The IP protocol at the IP destination address given in the datagram could not be reached. This is returned when the destination host does not support the protocol you requested, or when the destination protocol is currently busy and cannot accept your datagram. Your application should retry the request occasionally. If the attempt fails repeatedly you should check that the protocol is correct, and then contact the administrator of your site or of the destination host. This code is returned in response to an ICMP destination unreachable message received for your application.

27 Unreachable port.

The destination port of the upper-level protocol (ULP) is not reachable. The port is specific to the upper-level protocol (for example, the UDP port, or the TCP port). The destination port is busy, or the destination host does not support the port. The handling of this code depends upon the protocol. This code is returned in response to an ICMP destination unreachable message received for your application.

28 Unable to send datagram without fragmenting.

This code is returned when a previously sent datagram was discarded because IP was forced to fragment the datagram, but the Don't Fragment flag was set in the IP header. Your application must send smaller datagrams, use a different route to the destination that allows larger datagrams, or permit IP to fragment the datagram. This code is returned in response to an ICMP destination unreachable message received for your application.

29 Incorrect option.

One of the options specified in the IP header of a previous data request was incorrectly formatted. This code may be returned in response to an ICMP parameter problem message received for your application.

30 Strict source route failed.

The route specified in a strict source route option could not be honored, and so the datagram was discarded. You should check the addresses and ensure that the route is viable. This may be a temporary condition, for example when one of the specified gateways is not operating or when one of the gateways does not have information about the next gateway in the source route. This code is returned in response to an ICMP destination unreachable message received for your application.

31 Timeout during datagram transit.

A datagram was discarded because the time-to-live field in the IP header reached zero. This can happen when: 1) the TTL field is too small for the number of networks that must be traversed to reach the destination, 2) the catenet (internet) is congested and the datagram spent more time in gateway queues than permitted by the TTL field, or 3) the datagram was caught in a routing loop and could not be routed to the final destination. You should increase the TTL if it is too small, or wait until the congestion or routing loop in the network has cleared. This code is returned in response to an ICMP time exceeded message received for your application.

32 Timeout during reassembly.

A datagram was discarded because not all fragments of the datagram were received within the time-to-live specified in the IP header. This can happen when a datagram is fragmented and then some of the fragments are discarded in transit. It can also happen if the TTL field is too small for the amount of time required to transmit all of the fragments through the internet. This code is returned in response to an ICMP time exceeded message received for your application.

33 Insufficient memory in CDCNET device interface.

The CDCNET Device Interface where IP and IP Application Gateway resides is extremely congested and has run out of memory. The application should retry the request later.

34 Datagram discarded due to congestion.

A datagram may have been discarded due to congestion in CDCNET, in an intermediate network, in the destination network, or in the destination host. This code can be returned in response to an ICMP source quench message received for your application.

35 Incorrect ICMP message type.

The type specified in an ICMP message request was incorrect. Check the type against the values permitted by the ICMP request.

36 Incorrect ICMP message code.

The code specified in an ICMP message request was incorrect for the specified message type. Check the code against the values permitted for the ICMP message type.

37 IP internal error.

This code is returned when an internal error has occurred within CDCNET, IP, or the IP application gateway. Your application should abort and take a dump. Check the application carefully; it is possible that part of the IP routines is being overwritten by an overindexed array. If the application appears correct, you should refer the problem to your site analyst, or should write a Programming System Report (PSR) and notify your CDC representative.

19.2.7 Wait for Reply (WAIT)

When your application issues an IP request, the request is encapsulated by the IP interface routines, and sent via NOS Network Access Method (NAM) to a CDCNET process called the IP Gateway. The IP Gateway takes the encapsulated request and issues it to the actual IP process in CDCNET. The IP process returns a reply which the IP Gateway encapsulates and sends back to the IP interface routines. These routines take the encapsulated reply and return it to you in the request reply variable.

Since the request and reply must traverse a network (or at least, must move from the CYBER to a CDCNET MDI), there can be a delay of approximately 200 milliseconds (sometimes more) between the time the request is issued and the time the reply is received. For many communications applications, it is not feasible to wait this entire time while unable to perform any other task.

Most IP requests have a type logical parameter called 'wait'. If this parameter is set to TRUE, then the routine will not return until the reply has been received. If this parameter is set to FALSE, then the request will be sent to CDCNET for processing and then the routine will return immediately, before the reply has been received. The reply will be received at some later time by calls to the IP accept indication/reply (IPAI) routine.

During the period between the time the request is sent, and the time the reply is received, your application can continue to issue network requests and process data. Any return (output) parameters, such as addresses and identifiers, cannot be used or referenced. Your application cannot use the reply variable supplied in the request. This is extremely important, because the IP interface routines use this reply variable to hold information used to match incoming replies with incomplete requests. If the application mistakenly changes this value before the reply is received, then the IP interface will be unable to return the reply when it is received, and will be forced to abort the SAP associated with the incomplete request. The application can determine when the request has completed and the reply is returned by periodically examining the value of the reply variable. As long as the reply is negative, the request is incomplete and the reply variable cannot be changed or re-used. When the reply variable becomes positive (this will happen during an IPAI call), then the request is complete and the reply variable can be changed or re-used. After completion, any other return parameters are defined and can then be referenced and re-used.

It is recommended that your application use wait=TRUE only for initialization and termination. The application should use wait=FALSE in most other cases, since most IP applications must multiplex the sending and receiving of datagrams in a timely or real-time manner.

When wait=FALSE is in use, it is important to call IPAI frequently, preferably at least once for each network request issued. This is typically done in the form of a polling loop, where IPAI is called at the beginning of the loop and then some process is performed depending upon the result.

Most network processes must frequently wait for some activity before they are able to proceed. To avoid wasting host resources while waiting when wait=FALSE is in use, the processing loop should include a call to the routine DDNIDLE when no activity is returned from the IPAI call.

When wait=FALSE is used, there will be times when network resources are all busy with outstanding requests and temporarily cannot accept any more requests. In these cases, your application must be prepared to wait and retry the requests at a later time.

19.3 Subroutine Description Format

The IP interface subroutines listed in this chapter are described using the following format:

[1] Brief description of the subroutine's purpose.

[2] Subroutine calling format. The subroutine FORTRAN calling sequence is shown with a list of parameters. If the word CALL is used, then the routine is an IP request supplied by the IP interface and called by the application. If the word SUBROUTINE is used, then the routine is an IP indication supplied by the application and called by the TCP interface.

[3] Description of each parameter. The parameter name is listed followed by the FORTRAN TYPE (INTEGER, LOGICAL, CHARACTER*n, ARRAY) of the parameter and its mode (INPUT, OUTPUT or INPUT/OUTPUT). A textual description follows that includes range descriptions and the meaning of specific values.

[4] Remarks, including restrictions, rules, references to other manuals, or other special information about the command.

19.4 IPAI - IP Accept Indication/Status

Purpose	IP request polls for the delivery of an IP indication or of a reply to an incomplete IP request.
Format	CALL IPAI (user_sapid, ip_sapid, buffer, buffer_length, offset, status, wait)

Parameters user_sapid [INTEGER, OUTPUT]

Specifies the user service acess-point identifier that the application wishes to assign to this IP protocol. This identifier is supplied to the application as a parameter in subsequent indications for this SAP. The application is free to assign any 60-bit identifier of any type, though it will typically be an index into a protocol array, or some other integer that quickly distinguishes one protocol from another. The identifier must be unique, although IP does not check or enforce this. If the user_sapid is not unique, the application will not have any means to associate indications with specific SAPs.

ip_sapid [INTEGER, INPUT]

Identifies the IP SAP to poll for an indication. Its value can be either -1 or a specific IP service access-point identifier that was returned by a previous call to open an IP SAP (IPOS). If ip_sapid is -1, then an indication or reply for any IP SAP can be returned. If ip_sapid is a specific IP SAPID, then only the following will be returned:

1) Replies for incomplete requests for the SAP.
2) Indications for the SAP.
3) Close SAP and Error indications (IPCS and IPEI) for any SAP.

This parameter can be used primarily to constrain the receipt of data indications to a single SAP for short periods of time, or to allow an application to poll SAPs individually for data.

buffer [ARRAY, INPUT/OUTPUT]

Specifies the buffer to be used by the IP interface for receiving indications and replies from the network. This buffer must be large enough to contain the length specified by buffer_length.

buffer_length [INTEGER, INPUT]

Specifies the number of 8-bit bytes that can be accommodated in the buffer. It is recommended that you always use the NAM maximum block size, which is 2043 bytes. Using a larger value will not help, and use of a smaller value could cause truncation of an indication, which in turn aborts the SAP.

offset [INTEGER, OUTPUT]

Identifies the offset from the beginning of the buffer to the data that is passed in a data indication (IPDI or IPII). In some cases, the application data indication handler may be unable to process the data within the handler and may not wish to copy the data to another buffer for later processing due to the inherent inefficiency. In this case, the indication handler can provide some signal that data is available, and then the data can be extracted directly from the IPAI buffer, or the buffer can be queued, when the IPAI call returns. Sometimes, the data is prefaced by a header array that is used by the IP Interface and IP Gateway, so that the data does not begin at word 1 of the buffer array. This header is not passed to the indication handler, but is still present in the IPAI buffer.

The offset variable specifies where the data actually begins, where a zero offset specifies that the data is at the beginning of the buffer. That is, when IPAI returns after delivering a data indication, the parameter buffer (1+offset) represents exactly the same memory location as the IPDI/IPII parameter data (1).

The value of offset has no meaning if a non-data indication or a reply is delivered by the IPAI call.

status [INTEGER, OUTPUT]

SANDIN SAYS: "IT'S WRONG."

Returns the status of the IPAI request. If the code is negative then you issued the request with wait=FALSE and the request has not completed; you should check the value of 'status' periodically after calls to IPAI until a positive return code is received. When the request is complete or has been rejected, one of the following status codes will be returned:

Code	Description
1	No indications or status returned.
4	Application did not call DDNRA first.
5	Network unavailable; the network link has been lost.
9	Indication delivered.
10	Request status delivered.
11	The supplied status variable is already in use.
23	SAP does not exist, incorrect IP SAPID.
37	IP internal error.

Refer to the common parameters subsection above for a complete description of the status codes and their meaning.

wait [LOGICAL, INPUT]

Logical variable should be set to TRUE if the application wishes to wait for an indication or reply to be delivered. If the application wishes to continue execution while waiting for indications or replies to previous requests, then the parameter should be set to FALSE.

Remarks

The application must call IPAI frequently to poll for indications on SAPs. If the application is issuing IP requests with wait=FALSE then IPAI must be called frequently to receive the replies when the requests complete. It is recommended that the application call IPAI approximately once per IP request called, especially if wait=FALSE is in use. When the application is designed as a table-driven queue processor (or equivalent main-loop style of execution), then IPAI can be called as one of the event generators.

When you call IPAI with wait=TRUE and with ip_sapid set to a specific SAP identifier, it is possible that the indication or reply that is delivered will not be for the specified SAP. This is because certain indications (IPCS and IPEI) may be delivered regardless of the ip_sapid setting. If your application calls IPAI to poll a specific SAP, you are guaranteed that no data indications will be delivered for any other SAP.

19.5 IPCS - IP Close SAP

Purpose

IP request closes the application's IP service access point (SAP) for the specified IP protocol.

Format	CALL IPCS (protocol, ip_sapid, status, wait)

Parameters

protocol [INTEGER, INPUT]

Identifies the IP protocol associated with the SAP to close. Protocol numbers are assigned by SRI, and are listed in RFC-790 ASSIGNED NUMBERS.

ip_sapid [INTEGER, INPUT]

Identifies the IP SAP that you wish to close. This is the IP service access point identifier assigned in a previous call to open the IP SAP (IPOS). The ip_sapid and protocol parameters must both match those given in an IPOS request.

status [INTEGER, OUTPUT]

Returns the status of the close request. If the code is negative then you issued the request with wait=FALSE and the request has not completed; you should check the value of 'status' periodically after calls to IPAI until a positive return code is received. When the request is complete or has been rejected, one of the following status codes will be returned:

Code	Description
20	Successful request.
2	Previous operation incomplete.
3	Resources busy, try again later.
4	Application did not call DDNRA first.
5	Network unavailable; the network link has been lost.
11	The supplied status variable is already in use.
22	Incorrect protocol.

Code	Description
23	SAP does not exist, incorrect IP SAPID.
37	IP internal error.

Refer to the common parameters subsection above for a complete description of the status codes and their meaning.

wait [LOGICAL, INPUT]

Logical variable should be set to TRUE if the application wishes to wait for the reply to the request. If the application wishes to continue execution while waiting for the reply, then the parameter should be set to FALSE. Refer to the common parameters subsection above for a description of the purpose and use of the 'wait' parameter.

Remarks

A close SAP request is normally issued as part of application termination processing. Once all SAPs have been closed, no IP requests other than an IP open SAP request (IPOS) can be issued.

Once a SAP is closed, the application will no longer receive IP/ICMP indications for the specified IP protocol.

A close SAP indication (IPCSI) is not delivered when IPCS is called.

19.6 IPDI - IP Data Indication

Purpose

User-supplied IP indication handler is called by IP to deliver datagrams for the SAP.

Format CALL IPDI (ip_header, source_addr, dest_addr, ip_options, user_sapid, data, data_length)

Parameters ip_header [ARRAY, INPUT]

Identifies the IP header fields in the datagram. This is a 21-word array. Refer to the common parameters subsection above for a complete description of the ip_header parameter.

source_addr [ARRAY, INPUT]

Identifies the IP source address of the datagram. This is a 3-word array containing the in-use status, network address, and host address. Refer to the common parameters subsection above for a complete description of the source_addr parameter.

dest_addr [ARRAY, INPUT]

Identifies the IP destination address of the datagram. This is a 3-word array containing the in-use status, network address, and host address. Refer to the common parameters subsection above for a complete description of the dest_addr parameter.

ip_options [ARRAY, INPUT]

Identifies the IP options in the datagram. This is a 39-word array. Refer to the common parameters subsection above for a complete description of the ip_options parameter.

user_sapid [INTEGER, INPUT]

Identifies the user sapid as previously specified in an IP open SAP request (IPOS). The application subroutine uses this to identify the SAP and associated IP protocol of the datagram.

data [ARRAY, INPUT]

An array containing data that was sent by the peer application. The data consists of 8-bit bytes packed 7.5 bytes per 60-bit word. The first bit of the first byte of data starts in the uppermost bit (COMPASS bit 0) of the first word, and the subsequent bytes follow sequentially from there. When the number of bytes received does not fill an integral number of 60-bit words, the trailing, unused bits of the last word should be ignored.

The array that is passed is a subset of the array the application supplied in the IPAI call. The application can use this fact to process the data after the IPAI call, without the need to copy the data, if it is unable to process the data within the indication handler. Refer to the subsection above on the IPAI request for more information.

data_length [INTEGER, INPUT]

Identifies the number of 8-bit bytes in the data buffer. This parameter is zero if no data was delivered; it is possible to receive null data indications. The maximum value the application will receive is 2043 bytes.

Remarks

Datagram indications are delivered only for protocols for which the application has open SAPs.

The nature of the IP protocol is such that you may not receive some datagrams sent by the peer application, you may receive datagrams in a different order, and you may receive duplicate datagrams. The upper-level protocol must provide mechanisms such as retransmission and sequence numbers to account for this.

19.7 IPEI - IP Error Indication

Purpose

User-supplied IP indication handler is called by IP to notify the application that an error, ICMP error message, or unusual event has occurred for the specified IP protocol.

Format

CALL IPEI (ip_header, source_addr, dest_addr, ip_options, user_sapid, data, data_length, status, bad_param)

Parameters

ip_header [ARRAY, INPUT]

Identifies the IP header fields in the erroneous datagram or ICMP error message. This is a 21-word array. Refer to the common parameters subsection above for a complete description of the ip_header parameter.

source_addr [ARRAY, INPUT]

Identifies the IP source address of the erroneous datagram or ICMP error message. This is a 3-word array containing the in-use status, network address, and host address. Refer to the common parameters subsection above for a complete description of the source_addr parameter.

dest_addr [ARRAY, INPUT]

Identifies the IP destination address of the erroneous datagram or ICMP error message. This is a 3-word array containing the in-use status, network address, and host address. Refer to the common parameters subsection above for a complete description of the dest_addr parameter.

ip_options [ARRAY, INPUT]

Identifies the IP options in the erroneous datagram or ICMP error message. This is a 39-word array. Refer to the common parameters subsection above for a complete description of the ip_options parameter.

user_sapid [INTEGER, INPUT]

Identifies the user sapid as previously specified in an IP open SAP request (IPOS). The application subroutine uses this to dentify the SAP and associated IP protocol of the error indication.

data [ARRAY, INPUT]

An array containing the data in the erroneous datagram, or the data supplied with an ICMP error message. The data consists of 8-bit bytes packed 7.5 bytes per 60-bit word. The first bit of the first byte of data starts in the uppermost bit (COMPASS bit 0) of the first word, and the subsequent bytes follow sequentially from there. When the number of bytes received does not fill an integral number of 60-bit words, the trailing, unused bits of the last word should be ignored.

The array that is passed is a subset of the array the application supplied in the IPAI call. The application can use this fact to process the data after the IPAI call, without the need to copy the data, if it is unable to process the data within the indication handler. Refer to the subsection above on the IPAI request for more information.

data_length [INTEGER, INPUT]

Specifies the number of 8-bit bytes in the data buffer. This parameter is zero if no data was delivered with the data indication. The maximum value the application will receive is 2043 bytes.

status [INTEGER, ~~OUTPUT~~] INPUT

Specifies ~~Returns~~ the reason for the error indication. One of the following codes will be ~~returned~~ Supplied:

Code	Description
6	SAP inactive.
12	Network idledown in progress.
22	Incorrect protocol.
24	Unreachable network address.
25	Unreachable host address.
26	Unreachable protocol.
27	Unreachable port.
28	Unable to send datagram without fragmenting.
29	Incorrect option.
30	Strict source route failed.
31	Timeout during datagram transit.
32	Timeout during reassembly.
33	Insufficient memory in CDCNET device interface.
34	Datagram discarded due to congestion.
37	IP internal error.

Refer to the common parameters subsection above for a complete description of the status codes and their meaning.

bad_param [INTEGER, INPUT]

An integer that identifies an invalid IP header value. It is an index into the IP header, beginning with 0. This parameter has meaning only when the status code is 29.

Remarks An error indication is returned if IP immediately discarded a datagram issued by a previous send request (IPSR).

An error indication is also returned when an ICMP error message is received for the protocol associated with your IP SAP. The ICMP messages that generate error indications are Destination Unreachable, Source Quench, Time Exceeded, and Parameter Problem.

19.8 IPII - IP ICMP Indication

Hdr ?

Purpose User-supplied IP indication handler is called by IP to deliver general-interest ICMP message, and ICMP messages that belong specifically to the protocol associated with the application SAP.

Format CALL IPII (source_addr, dest_addr, icmp_header, user_sapid, data, data_length)

Parameters source_addr [ARRAY, INPUT]

Identifies the IP source address of the ICMP message. This is a 3-word array containing the in-use status, network address, and host address. Refer to the common parameters subsection above for a complete description of the source_addr parameter.

dest_addr [ARRAY, INPUT]

Identifies the IP destination address of the ICMP message. This is a 3-word array containing the in-use status, network address, and host address. Refer to the common parameters subsection above for a complete description of the dest_addr parameter.

icmp_header [ARRAY, INPUT]

Identifies the ICMP header fields of the ICMP message. This is a 4-word array. Refer to the common parameters subsection above for a complete description of the icmp_header parameter.

user_sapid [INTEGER, INPUT]

Identifies the user sapid as previously specified in an IP open SAP request (IPOS). The application subroutine uses this to identify the SAP and associated IP protocol of the ICMP message.

data [ARRAY, INPUT]

An array containing the data supplied with the ICMP message. The data consists of 8-bit bytes packed 7.5 bytes per 60-bit word. The first bit of the first byte of data starts in the uppermost bit (COMPASS bit 0) of the first word, and the subsequent bytes follow sequentially from there. When the number of bytes received does not fill an integral number of 60-bit words, the trailing, unused bits of the last word should be ignored.

The array that is passed is a subset of the array the application supplied in the IPAI call. The application can use this fact to process the data after the IPAI call, without the need to copy the data, if it is unable to process the data within the indication handler. Refer to the subsection above on the IPAI request for more information.

Refer to the common parameter subsection above for the meaning and format of ICMP message data, which varies depending upon the message type.

data_length [INTEGER, INPUT]

Specifies the number of 8-bit bytes in the data buffer. This parameter is zero if no data was delivered with the data indication. The maximum value the application will receive is 2043 bytes.

Remarks

The following ICMP messages are delivered to all IP applications: Destination Unreachable, Source Quench, Redirect, Time Exceeded.

The following ICMP messages are delivered only to the initiating application, as determined by the IP protocol embedded in the upper 8-bits of the message identifier: Echo Reply, Time Reply, Information Reply.

The following ICMP messages are never delivered by an ICMP indication, but instead are delivered via an error indication to the application that caused the error: Parameter Error.

The following ICMP messages are handled internally by IP and are never delivered to an application: Echo Request, Time Request, and Information Request.

19.9 IPIR - IP ICMP Message Request

Purpose IP request sends an ICMP message to the specified destination.

Format CALL IPIR (source_addr, dest_addr, icmp_header, protocol, ip_sapid, data, data_length, status, wait)

Parameters source_addr [ARRAY, INPUT]

Identifies the IP source address of the ICMP message. This is a 3-word array containing the in-use status, network address, and host address. Refer to the common parameters subsection above for a complete description of the source_addr parameter.

dest_addr [ARRAY, INPUT]

Identifies the IP destination address of the ICMP message. This is a 3-word array containing the in-use status, network address, and host address. Refer to the common parameters subsection above for a complete description of the dest_addr parameter.

icmp_header [ARRAY, INPUT]

Identifies the ICMP header fields of the ICMP message to send. This is a 4-word array. Refer to the common parameters subsection above for a complete description of the icmp_header parameter.

protocol [INTEGER, INPUT]

Specifies the IP protocol to use with the ICMP message. This has meaning only for Echo, Timestamp, and Information Request messages, where IP uses the protocol as a part of the message identifier so that it can return the message reply to the correct application. You must have an IPSAP opened for the specified protocol to receive the reply.

ip_sapid [INTEGER, INPUT]

Identifies the IP SAP associated with the ICMP message. This is the IP service access point identifier assigned in a previous call to open the IP SAP (IPOS).

data [ARRAY, INPUT]

An array containing data to be sent with the ICMP message. The data consists of 8-bit bytes packed 7.5 bytes per 60-bit word. The first bit of the first byte of data starts in the uppermost bit (COMPASS bit 0) of the first word, and the subsequent bytes follow sequentially from there. When the number of bytes sent does not fill an integral number of 60-bit words, the trailing, unused bits of the last word are ignored.

Refer to the common parameter subsection above for the meaning and format of ICMP message data, which varies depending upon the message type.

data_length [INTEGER, INPUT]

Specifies the number of 8-bit bytes in the data buffer. This parameter is zero if no data is to be delivered with the ICMP request. The maximum value the application can send is 2043 bytes.

status [INTEGER, OUTPUT]

Returns the status of the ICMP request. If the code is negative then you issued the request with wait=FALSE and the request has not completed; you should check the value of 'status' periodically after calls to IPAI until a positive return code is received. When the request is complete or has been rejected, one of the following status codes will be returned:

Code	Description
20	Successful request.
2	Previous operation incomplete.
3	Resources busy, try again later.
4	Application did not call DDNRA first.
5	Network unavailable; the network link has been lost.
11	The supplied status variable is already in use.
23	SAP does not exist, incorrect IP SAPID.
24	Unreachable network address.
25	Unreachable host address.
33	Insufficient memory in CDCNET device interface.
35	Incorrect ICMP message type.
36	Incorrect ICMP message code.
37	IP internal error.

Refer to the common parameters subsection above for a complete description of the status codes and their meaning.

wait [LOGICAL, INPUT]

Logical variable should be set to TRUE if the application wishes to wait for the reply to the request. If the application wishes to continue execution while waiting for the reply, then the parameter should be set to FALSE. Refer to the common parameters subsection above for a description of the purpose and use of the 'wait' parameter.

Remarks — Care should be taken when sending ICMP messages,as these may affect the operation of other hosts on the network.

19.10 IPOS - IP Open SAP

Purpose — IP request opens an IP service access point so that the application can use IP services. A SAP must be opened before any other IP request can be issued, and a SAP must be opened for each IP protocol for which the application wishes to receive data.

Format — CALL IPOS (protocol, user_sapid, ip_sapid, status, wait)

Parameters — protocol [INTEGER, INPUT]

Identifies the IP protocol associated with the SAP to open. Protocol numbers are assigned by SRI, and are listed in RFC-790 ASSIGNED NUMBERS. This is the default protocol used datagram headers for this SAP when no header fields are specified. It is also the protocol used for all indications.

user_sapid [INTEGER, INPUT]

Specifies the user service access point (SAP) identifier that the application wishes to assign to this IP SAP. This identifier is supplied to the application as a parameter in subsequent indications for this SAP. The application is free to assign any 60-bit identifier of any type, though it will typically be an index into a protocol array, or some other integer that quickly distinguishes one SAP from another. The identifier must be unique, although IP does not check or enforce this. If the user_sapid is not unique, the application will not have any means to associate indications with specific SAPs.

ip_sapid [INTEGER, OUTPUT]

Identifies the IP service for subsequent IP requests. This is the IP service access point identifier assigned by the IP interface. It must be saved and supplied in any subsequent requests to IP for the specified protocol.

status [INTEGER, OUTPUT]

Returns the status of the open request. If the code is negative then you issued the request with wait=FALSE and the request has not completed; you should check the value of 'status' periodically after calls to IPAI until a positive return code is received. When the request is complete or has been rejected, one of the following status codes will be returned:

Code	Description
20	Successful request.
2	Previous operation incomplete.
3	Resources busy, try again later.
4	Application did not call DDNRA first.
5	Network unavailable; the network link has been lost.
11	The supplied status variable is already in use.
21	Protocol already in use.
22	Incorrect protocol.
33	Insufficient memory in CDCNET device interface.
37	IP internal error.

Refer to the common parameters subsection above for a complete description of the status codes and their meaning.

wait [LOGICAL, INPUT]

Logical variable should be set to TRUE if the application wishes to wait for the reply to the request. If the application wishes to continue execution while waiting for the reply, then the parameter should be set to FALSE. Refer to the common parameters subsection above for a description of the purpose and use of the 'wait' parameter.

Remarks None.

19.11 IPSR - IP Send Data Request

Format CALL IPSR (ip_header, source_addr, dest_addr, ip_options, ip_sapid, data, data_length, status, wait)

Parameters ip_header [ARRAY, INPUT]

Specifies the IP header fields in the datagram. This is a 21-word array. Refer to the common parameters subsection above for a complete description of the ip_header parameter. Most typical applications will not specify any IP header fields; in this case, a zero can be used for the parameter and it is not necessary to allocate an array.

source_addr [ARRAY, INPUT]

Identifies the IP source address of the datagram. This is a 3-word array containing the in-use status, network address, and host address. Refer to the common parameters subsection above for a complete description of the source_addr parameter.

dest_addr [ARRAY, INPUT]

Identifies the IP destination address of the datagram. This is a 3-word array containing the in-use status, network address, and host address. Refer to the common parameters subsection above for a complete description of the dest_addr parameter.

ip_options [ARRAY, INPUT]

Specifies the IP options in the datagram. This is a 39-word array. Refer to the common parameters subsection above for a complete description of the ip_options parameter. Most typical applications will not specify any IP option fields; in this case, a zero can be used for the parameter and it is not necessary to allocate an array.

ip_sapid [INTEGER, INPUT]

Identifies the IP SAP associated with the datagram. This is the IP service access point identifier assigned in a previous call to open the IP SAP (IPOS). The SAP determines the protocol to be used in the IP header if none is specified by the ip_header parameter.

data [ARRAY, INPUT]

An array containing the data to be sent in an IP datagram. The data consists of 8-bit bytes packed 7.5 bytes per 60-bit word. The first bit of the first byte of data starts in the uppermost bit (COMPASS bit 0) of the first word, and the subsequent bytes follow sequentially from there. When the number of bytes sent does not fill an integral number of 60-bit words, the trailing, unused bits of the last word are ignored.

data_length [INTEGER, INPUT]

Specifies the number of 8-bit bytes in the data buffer. This parameter should be set to zero if no data is to be sent. Null data blocks are possible, and even useful in conjunction with options and header data. The largest possible datagram length is 2043 bytes.

status [INTEGER, OUTPUT]

Returns the status of the data request. If the code is negative then you issued the request with wait=FALSE and the request has not completed; you should check the value of 'status' periodically after calls to IPAI until a positive return code is received. When the request is complete or has been rejected, one of the following status codes will be returned:

Code	Description
20	Successful request.
3	Resources busy, try again later.
4	Application did not call DDNRA first.
5	Network unavailable; the network link has been lost.
11	The supplied status variable is already in use.
37	IP internal error.

Refer to the common parameters subsection above for a complete description of the status codes and their meaning. If IP or IP Gateway must reject the datagram request, this is indicated via an error indication (IPEI) and not via the status reply.

wait [LOGICAL, INPUT]

Logical variable should be set to TRUE if the application wishes to wait for NAM to acknowledge the data. If the application wishes to continue execution while waiting for the ACK, then the parameter should be set to FALSE. Refer to the common parameters subsection above for a description of the purpose and use of the 'wait' parameter, but there are some differences. Any rejection of the data request will occur immediately; that is, any failure response will be returned when IPSR returns. When data is sent, the NAM block limit (data window) is reduced, and wait=TRUE can be used to wait for NAM to restore the block limit (re-open the window). The use of wait=TRUE also allows unlimited block size; refer to the data_length parameter.

Remarks

For efficiency, IP gateway does not return a reply to data requests like it does for other requests, since these replies would use bandwidth. Any negative response from IP for a data request will generate an error indication (IPEI).