DIAGNOSTIC SUITE
TECHNICAL REFERENCE

51 E Campbell Ave
Suite 160
Campbell, CA. 95008

email: support@iniche.com
http://www.iniche.com
support: 408.540.1160
fax: 408.540.1161

InterNiche Technologies Inc. has made every effort to assure the accuracy of the information contained in this documentation. We appreciate any feedback you may have for improvements. Please send your comments to **support@iniche.com**.

Rev-10.2005

TABLE OF CONTENTS

1. OVERVIEW

This document describes how to use InterNiche Technologies' Diagnostic Suite, our standard debugging, monitoring, and runtime configuration commands for our TCP/IP stack, related protocols, and applications. The InterNiche Diagnostic Interface consists of three elements:

? Instrumentation (or Agents) - built into all InterNiche products
? User Display - the command line interface to the Diagnostic environment
? Menu System - an online list of, and help for, our Diagnostic commands

The InterNiche Diagnostic Suite is a command line driven interface and requires that the InterNiche TCP/IP stack software be installed and operational in order to function properly.

2. COMMAND DESCRIPTIONS

This section describes the various line commands. They are organized into four areas of utility: **General** commands, **Diagnostic** commands, **Protocol** specific commands, and **Target** specific commands. Note that the Protocol specific commands are available only if you have licensed and included the additional protocols from InterNiche as a part of your system. NATRouter, RIP, FTP, TELNET, SNMP, Emailer, WebPort, and DHCP are examples of optional protocols, and their related diagnostic commands are available when they have been built into your executable.

2.1 The InterNiche Console

When **webport.exe** is run, it initializes and gives the **INET>** prompt. We call this the InterNiche console. The console is the computer on which **webport.exe** is executing. The line commands can be displayed as Help menu lists or simply entered directly at the prompt.

The device on which this prompt is displayed is determined by the porting engineer and will vary depending on the nature of the target system. For our DOS demo, the prompt is displayed on the target PC's monitor and input from the user is accepted from the PC's keyboard. On target system's without a keyboard and monitor, the prompt could be output on a serial port and user input would be accepted on the serial port. For purpose of this discussion we will refer to this user interface input/output device as the target system's "console".

2.2 Runtime Commands

Commands are available in **webport.exe** when the option **IN_MENUS** is defined in the file **ipport.h_h.** The menuing code is portable to systems which opt to use the InterNiche console menuing system and incorporate the InterNiche TCP/IP stack.

The InterNiche user interface prompts the user with the following prompt:

```
INET>_
```

At this point the user can type any of dozens of commands for execution by the agent. Our command interface is a "smart" interface and only requires that you enter enough of the command name to uniquely identify it. Many commands can also take 1 or more parameters typed on the command line.

The first two sections cover the General commands, Diagnostic commands. These should help a new user get started quickly. The remainder of these sections explain each command in detail in the order that they appear in the menus.

As an introduction to these commands, an annotated session transcript is reproduced here. The command as issued is shown in **bold**; a convention that is followed throughout this document. Some commonly used commands are **host** and **ping**.

```
INET> host 10.0.0.5
active host number set to 10.0.0.5
```

The **host** command sets a default remote host IP address for those commands that require a remote host to be specified.

```
INET> ping
ping 0 to 10.0.0.5: ping 1 sent…
got ping reply; len:62 seq 0 from 10.0.0.5
```

The **ping** command checks if a host is reachable or not.

```
INET> q
```

The **quit** command is invoked (abbreviated to **q**) and the station returns to DOS.

Parameters can be abbreviated to as few as a single letter when the meaning is unambiguous.

3. GENERAL COMMANDS

3.1 Help

The Help menu for the InterNiche console displays a list of available commands. The commands can be brought up on the display by entering **help** or a question mark, **?**. The **help** command by itself displays the general commands. When given **diagnostic** as an argument, it lists the diagnostic command set.

The "`Also try`" line at the bottom of the Help menu lists the other InterNiche modules that are installed and for which diagnostic help is available.

COMMAND

help
?

PARAMETERS

general	Displays the most commonly used commands
diagnostic	Displays commands that are used for reporting statistics
module name	The remaining parameters in the help list will vary according to which options were included in your build.

EXAMPLE

```
INET> help
SNMP Station: general commands:
   help        - help with menus
   state       - show current station setup
   quit        - quit station program
   nvset       - set non-volatile parameters
   version     - display version information
   !command    - pass command to OS shell
Also try 'help [general|diagnostic|nat|ping]'
```

DESCRIPTION

This command is used to list all the other commands. When typed by itself, the general commands are listed. When followed by **diagnostic** or a specific configuration name the appropriate command sets are listed. Similarly, **help nat** would display the commands for NATRouter.

A question mark, **?**. is the same as **help**.

Parameters can be abbreviated to as few as a single letter when the meaning is unambiguous.

3.2 General

COMMAND

state - show current setup

PARAMETERS

None

EXAMPLE

```
INET> state
Station IP address for iface 0: 192.9.200.1
Station IP address for iface 1: 127.0.0.1
Active remote host 127.0.0.1
Community string "public"
Object Id: 1.3.6.1.2.1.1.3.0
retry/ping delay time: 1008 ms.
session open to 127.0.0.1, ports: local-1205, remote-161(snmp)
session->timeout: 6 ticks. (tick is 1/18th sec)
No MIBs loaded from numbers files
INET> _
```

DESCRIPTION

Displays current state information, interfaces, and default settings.

COMMAND

quit - quit station program

PARAMETERS

None

EXAMPLE

```
INET> quit
c:\> _
```

DESCRIPTION

Quit will cause the application to do a clean exit to the OS, in this case DOS. It disconnects the InterNiche IP stack from the ODI and Packet drivers. The stack is removed from memory since it is part of the application.

COMMAND

version - display version information

PARAMETERS

None

EXAMPLE

```
INET> version
InterNiche's portable TCP/IP demo for DOS, v1.53
INET> _
```

DESCRIPTION

Displays current version of the InterNiche TCP/IP stack.

COMMAND

!*command* - pass command to OS shell

PARAMETERS

Any valid OS command

EXAMPLE

```
INET> !dir
Volume in drive C is MAIN DISK
 Volume Serial Number is 1E4D-17D0
 Directory of C:\

AUTOEXEC DOS           211  11-27-98 12:56p AUTOEXEC.DOS
COMMAND  COM        93,812  08-24-96 11:11a COMMAND.COM
CONFIG   DOS           188  11-27-98 12:56p CONFIG.DOS
AUTOEXEC BAT           301  12-04-98 12:25p AUTOEXEC.BAT
WINDOWS        <DIR>        11-23-98  4:15p WINDOWS
INFOS          <DIR>        12-02-98  7:55p INFOS
AUTOEXEC VIA            54  11-23-98  4:21p AUTOEXEC.VIA
PROGRA~1       <DIR>        11-23-98  4:15p Program Files
AUTOEXEC VIA            54  11-23-98  4:21p AUTOEXEC.VIA
SCANDISK LOG           518  01-11-99 10:34a SCANDISK.LOG
TEMP           <DIR>        12-29-98 12:26p TEMP
CONFIG   WIN           188  11-27-98  1:27p CONFIG.WIN
WINUTILS       <DIR>        11-27-98  1:26p WINUTILS
WINZIP         <DIR>        11-27-98  1:27p WinZip
CONFIG   SYS            47  12-03-98 12:13p CONFIG.SYS
ARCHIVES       <DIR>        11-27-98  1:42p Archives
AUTOEXEC 000           125  12-02-98  6:27p AUTOEXEC.000
MYDOCU~1       <DIR>        11-27-98  3:56p My Documents
AUTOEXEC BAK            53  11-30-98 12:23p AUTOEXEC.BAK
CONFIG   000             2  12-02-98  6:26p CONFIG.000
SCANDISK LOG           518  01-11-99 10:34a SCANDISK.LOG
TEMP           <DIR>        12-29-98 12:26p TEMP
CONFIG   WIN           188  11-27-98  1:27p CONFIG.WIN
WINUTILS       <DIR>        11-27-98  1:26p WINUTILS
WINZIP         <DIR>        11-27-98  1:27p WinZip
BIN            <DIR>        12-03-98 11:22a bin
PUBLIC         <DIR>        12-09-98  6:35p Public
        13 dir(s)        2,812.29 MB free
INET> !copy webport.nv webport.sav
1 file(s) copied
INET> _
```

DESCRIPTION

Any command preceded by an asterisk "**!**" will be treated as an OS command, and will be executed as on an OS prompt. For example, in DOS, "**!dir**" will executed as "**dir**" command on DOS prompt.

Note: If there is insufficient memory to execute the command, nothing will happen.

3.3 **UDP Echo** - define UDPSTEST in ipport.h_h

COMMAND

uesend - open UDP echo client, send UDP echo packet

PARAMETERS

None

EXAMPLE

```
INET> uesend
echo socket not open. Opening....
udp echo client is starting.
sending UDP echo 0 to 10.0.0.22
INET> host 10.20
INET> uesend
host changed, restarting client socket
udp echo client is starting.
sending UDP echo 0 to 10.0.0.20
INET> UDP echo reply; len:64, reply:0, Our send#:0
INET> Deleting idle UDP Echo Client.
INET> _
```

DESCRIPTION

This command will open a UDP Echo Client and send a UDP packet to the UDP Echo Server. The IP address of the UDP Echo Server is specified using the **host** command. If a UDP Echo Client socket connection is already open for this Server, it will be used.

By default an idle UDP Echo Client is deleted after 10 minutes. This is definable in **UDP_IDLE_TIMEOUT**.

This command is included in the menu list if **UDPSTEST** was defined in **ipport.h_h**.

SEE ALSO

delay

COMMAND

uesinit - start UDP echo server

PARAMETERS

None

EXAMPLE

```
INET> uesinit
udp echo server is starting.
INET> _
```

DESCRIPTION

This command will start the UDP Echo Server on the console, if it is not already running. If it is running, then nothing is done.

This command is included in the menu list if **UDPSTEST** was defined in **ipport.h_h**.

COMMAND

uechalt - close UDP echo client

PARAMETERS

None

EXAMPLE

```
INET> uechalt
udp echo - closing client socket
INET> _
```

DESCRIPTION

This command will close the UDP Echo Client socket connection. An UDP Echo Client sends packets to an UDP Echo Server. The server sends them back. This mechanism is used to test the functionality of UDP protocol.

A technical note over here is that multiple Client socket connections can be open on the console. One Client socket connection via the console interface. If TELNET Server is implemented in console, then another computer can do a TELNET connection to the console, and open a Client socket connection.

So, while only one UDP Echo Server can be running on the console, there could be multiple UDP Echo Clients.

This command is included in the menu list if **UDPSTEST** was defined in **ipport.h_h**.

COMMAND

ueshalt - close UDP echo server

PARAMETERS

None

EXAMPLE

```
INET> ueshalt
udp echo - closing server socket
INET> _
```

DESCRIPTION

This command will close the UDP Echo Server (running on the console), if it is running. If it is not running, then nothing is done.

This command is included in the menu list if **UDPSTEST** was defined in **ipport.h_h**.

COMMAND

uestats - UDP echo statistics

PARAMETERS

None

EXAMPLE

```
INET> uestats
Showing UDP Echo statistics.
   There are no Server connections.
   There are no Client connections.
INET> uesend
echo socket not open. Opening....
udp echo client is starting.
sending UDP echo 0 to 10.0.0.20
INET> UDP echo reply; len:128, reply:0, Our send#:0
INET> uesinit
udp echo server is starting.
INET> uestats
Showing UDP Echo statistics.
   There is one Server connection.
   Total pkts for Client 1: sent=1,rcvd=1
   Total Client connections=1.
INET> _
```

DESCRIPTION

Show the statistics for UDP Echoes completed.

This command is included in the menu list if **UDPSTEST** was defined in **ipport.h_h**.

3.4 TCP Echo - define TCP_ECHOTEST in ipport.h_h

COMMAND

tesend - open TCP echo client, send TCP echo packet

PARAMETERS

None

EXAMPLE

```
INET> host 10.20
INET> tesend
All TCP Echo Client connections are in use.
Please try at a later time.
INET> tesend
sending TCP echo 0 to 10.0.0.20
INET> TCP echo reply from:10.0.0.20, len:64, reply:0,Our send#:0
INET> Deleting idle TCP Echo Client.
INET> _
```

DESCRIPTION

This command will open a TCP Echo Client and send a TCP packet to the TCP Echo Server. The IP address of the TCP Echo Server is specified using the "host" command. If a TCP Echo Client connection is already open for this Server, it will be used.

FD_SETSIZE in **tcpport.h** defines the maximum number of open TCP connections on which **select()** can be done. TCP Echo Client uses **select()** which has a default value of 2. Therefore by default only 1 TCP Echo Client can be open as the other connection is used for TCP Echo Server.

By default an idle TCP Echo Client is deleted after 10 minutes. This is definable in **TCP_IDLE_TIMEOUT**.

This command is included in the menu list if **TCP_ECHOTEST** was defined in **ipport.h_h**.

SEE ALSO

delay

COMMAND

tesinit - start TCP echo server

PARAMETERS

None

EXAMPLE

```
INET> tesinit
tcp echo srv - starting.
INET> _
```

DESCRIPTION

This command starts the TCP Echo Server on the console. If it is already running, nothing is done. Normally TCP Echo Server and UDP Echo Server are started with the application. And closed when the application is quit. So this command would be sparsely used.

This command is included in the menu list if **TCP_ECHOTEST** was defined in **ipport.h_h**.

COMMAND

teshalt - close TCP echo server

PARAMETERS

None

EXAMPLE

```
INET> host 10.22
INET> teshalt
tcp echo srv - closing.
INET> _
```

DESCRIPTION

Close the TCP Echo Server (on the console) if it is running. If it is not running, nothing is done.

This command is included in the menu list if **TCP_ECHOTEST** was defined in **ipport.h_h**.

COMMAND

techalt - close TCP echo client

PARAMETERS

None

EXAMPLE

```
INET> techalt
Closing TCP Echo Client.
INET> _
```

DESCRIPTION

This command will close the TCP Echo Client connection. An TCP Echo Client sends packets to an TCP Echo Server. The server sends them back. This mechanism is used to test the functionality of TCP protocol.

A technical note over here is that multiple Client connections can be open on the console. One Client connection via the console interface. If TELNET Server is implemented in console, then another computer can do a TELNET connection to the console, and open a Client connection.

So, while only one TCP Echo Server can be running on the console, there could be multiple TCP Echo Clients.

This command is included in the menu list if **TCP_ECHOTEST** was defined in **ipport.h_h**.

COMMAND

testats - TCP echo statistics

PARAMETERS

None

EXAMPLE

```
INET> testats
Showing TCP Echo statistics.
   There are no Server connections.
   There are no Client connections.
INET> tesend
sending TCP echo 0 to 10.0.0.20
INET> TCP echo reply from:10.0.0.20, len:128, reply:0,Our send#:0
INET> tesinit
tcp echo srv - starting.
INET> testats
Showing TCP Echo statistics.
   There is one Server connection.
   Total pkts for Client 1: sent=1,rcvd=1
   Total Client connections=1.
INET> Deleting idle TCP Echo Client.
INET> _
```

DESCRIPTION

Show the statistics for the TCP Echo connections.

This command is included in the menu list if **TCP_ECHOTEST** was defined in **ipport.h_h**.

3.5 **SNMP** - define INCLUDE_SNMP in ipport.h_h

COMMAND

trap - send a test trap

PARAMETERS

None

EXAMPLE

```
INET> trap
trap sent
INET> _
```

DESCRIPTION

This command sends a SNMP (version 1) trap to the trap host.

This command is included in the menu list if **INCLUDE_SNMP** was defined in **ipport.h_h**.

3.6 NV Parameters - define INCLUDE_NVPARAMS in ipport.h_h

COMMAND

nvset - set non-volatile parameters

PARAMETERS

None

EXAMPLE

```
INET> nvset
INET> _
```

DESCRIPTION

This command is included in the menu list if **INCLUDE_NVPARAMS** was defined in **ipport.h_h**. On execution **nvset** saves the current configuration to **webport.nv**.

4. DIAGNOSTIC COMMANDS

When **IN_MENUS** is defined in **ipport.h_h**, the Diagnostic Command menu is available and the list will vary according to what other options have been defined in the file **ipport.h_h** for inclusion. If **NET_STATS** is defined, the CUI will include the ability to display certain statistics.

FILE

nrmenus.c

COMMAND

help diagnostic

PARAMETER

diagnostic

EXAMPLE

```
INET> ? diag
SNMP Station: diagnostic commands:
   arps        - display ARP stats and table
   debug       - set IP stack debug tracing
   dtrap       - try to hook debugger
   dump        - hexdump incoming packets
   iface       - display net interface stats
   linkstats   - display link layer specific stats
   memory      - list currently alloced memory
   trapsize    - set size for alloc() trap
   udp         - display UDP layer stats
   snmp        - display SNMP MIB counters
   upcall      - trace received packets
INET> _
```

DESCRIPTION

Used as a parameter for Help, this command displays a list of the diagnostic commands that return current statistical data concerning the features included in your build.

COMMAND

buffers - display allocated packet buffer statistics

PARAMETERS

None

EXAMPLE

```
INET> buffers
  PACKET    len  buffer    que data
6F79:0018, 1536, 6F79:0048 big  00 80 C8 E2 AF 2A 00 40 26 2D 01
6C 08 00 45
6F79:064E, 1536, 6F79:067E big  FF FF FF FF FF FF 00 80 C8 E2 B8
98 08 00 45
6F79:0C84, 1536, 6F79:0CB4 big  FF FF FF FF FF FF 00 80 C8 E2 B8
98 08 00 45
6F79:12BA, 1536, 6F79:12EA big  FF FF FF FF FF FF 00 00 F4 90 10
52 08 00 45
6F79:18F0, 1536, 6F79:1920 big  00 80 C8 E2 AF 2A 00 40 26 2D 01
6C 08 00 45
6F79:1F26, 1536, 6F79:1F56 big  00 80 C8 E2 AF 2A 00 40 26 2D 01
6C 08 00 45
6F79:255C, 1536, 6F79:258C big  FF FF FF FF FF FF 00 80 C8 E2 B8
98 08 00 45
6F79:2B92, 1536, 6F79:2BC2 big  FF FF FF FF FF FF 00 80 C8 E2 B8
98 08 00 45
6F79:31C8, 1536, 6F79:31F8 big  FF FF FF FF FF FF 00 80 C8 E2 B8
98 08 00 45
6F79:37FE, 1536, 6F79:382E big  FF FF FF FF FF FF 00 80 C8 E2 B8
98 08 00 45
6F79:3E34, 1536, 6F79:3E64 big  FF FF FF FF FF FF 00 80 C8 E2 B8
98 08 00 45
6F79:446A, 1536, 6F79:449A big  00 80 C8 E2 AF 2A 00 40 26 2D 01
6C 08 00 45
6F79:4AA0, 1536, 6F79:4AD0 big  FF FF FF FF FF FF 00 80 C8 E2 B8
98 08 00 45
6F79:50D6, 1536, 6F79:5106 big  00 80 C8 E2 AF 2A 00 40 26 2D 01
6C 08 00 45
6F79:570C, 1536, 6F79:573C big  00 80 C8 E2 AF 2A 00 40 26 2D 01
6C 08 00 45
6F79:5D42, 1536, 6F79:5D72 big  00 80 C8 E2 AF 2A 00 40 26 2D 01
6C 08 00 45
6F79:6378, 1536, 6F79:63A8 big  FF FF FF FF FF FF 00 80 C8 E2 B8
98 08 00 45
6F79:69AE, 1536, 6F79:69DE big  00 80 C8 E2 AF 2A 00 40 26 2D 01
6C 08 00 45
6F79:6FE4, 1536, 6F79:7014 big  FF FF FF FF FF FF 00 80 C8 E2 B8
98 08 00 45
....press any key for more (ESC to break)....
INET> _
```

DESCRIPTION

Shows statistics for the allocated buffers.

COMMAND

queues - dump packet buffer queues

PARAMETERS

None

EXAMPLE

```
INET> queues
bigfreeq: head:6F79:1F26, tail:6F79:18F0, len:30, min:2, max:30
lilfreeq: head:6F79:C8A4, tail:6F79:CB7C, len:29, min:0, max:30
rcvdq: head:0000:0000, tail:0000:0000, len:0, min:0, max:2
INET> _
```

DESCRIPTION

The first two lines provide tally information about the big and little packet buffer free queues. The **len** field gives an instantaneous snapshot of how many packet buffers of each type are in the queues. The **min** field tells you how low **len** has gotten, which gives you some indication of whether you're running out of packet buffers. When min is **0** it means that you've run out of packet buffers in the listed queue type at least once since you booted the stack.

The line about **rcvdq** is telling you how big the packet receive queue has gotten. **rcvdq len** tells you that you have packets in the receive queue that have not yet been processed by the IP layer. **rcvdq max** lets you know how high **len** has gotten to be since boot time. A high value in **rcvdq max** would be an indication that the stack isn't getting around to processing the receive queue in a timely manner.

COMMAND

dbytes - dump block of memory

PARAMETERS

Memory location (hex seg:offset), length (optional)

EXAMPLE

```
INET> dbytes
enter memory location in hex seg:offset form, followed by optional
length
INET> dbytes 6F79:0018, 128
4E 06 79 6F 48 00 79 6F 00 06 48 00 79 6F 00 03
00 00 00 00 C0 C2 78 5F 0A 00 00 50 08 00 00 00
00 00 00 00 00 00 0E 00 4D 00 06 06 4D 00 4D 4D
00 80 C8 E2 AF 2A 00 40 26 2D 01 6C 08 00 45 00
02 F2 08 1A 00 00 1E 06 7D 87 0A 00 00 16 0A 00
00 50 00 17 0B 5C 00 CD 34 07 24 DF 76 AD 50 18
20 00 9F F9 00 00 30 30 20 30 30 20 0D 0A 30 30
20 30 30 20 30 30 20 30 30 20 30 30 20 30 30 20
30 30 20 30 30 20 30 30 20 30 30 20 30 30 20 30
30 20 30 30 20 30 30 20 30 30 20 30 30 20 0D 0A
30 30 20 30 30 20 30 30 20 30 30 20 30 30 20 30
30 20 30 30 20 30 30 20 30 30 20 30 30 20 30 30
20 30 30 20 30 30 20 30 30 20 30 30 20 30 30 20
0D 0A 30 30 20 30 30 20 30 30 20 30 30 20 30 30
20 30 30 20 30 30 20 30 30 20 30 30 20 30 30 20
30 30 20 30 30 20 30 30 20 30 30 20 30 30 20 30
30 20 0D 0A 30 30 20 30 30 20 30 30 20 30 30 20
30 30 20 30 30 20 30 30 20 30 30 20 30 30 20 30
30 20 30 30 20 30 30 20
INET> _
```

DESCRIPTION

This command helps in debugging.

The location parameter will be in seg:offset form only on processors that have the segmented memory system, i.e. 16 bit Intel x86 targets.

On 68K and RISC targets, the location will just be a single 32 bit number that indicates the address. This is also be true for Intel x86 targets that are running in protected mode (like DOS4GW).

COMMAND

debug - set IP stack debug tracing

PARAMETERS

Optional bitmask

EXAMPLE

```
INET> debug
IP stack debug tracing on
INET> debug
IP stack debug tracing off
INET> _
```

DESCRIPTION

This diagnostic command sets an internal flag which results in the application printing out scores of status messages as packets are received or sent up and down the protocol stack. This can be helpful for detecting at exactly what protocol stack layer a bad packet's error occurs. Often the nature of the error will be reported, i.e. "bad cksum".

The **debug** command accepts an optional parameter, which is a hex number that represents a bit mask in which each bit specifies whether a particular type of IP stack tracing will occur. Below are the definitions of the various bits:

BUGHALT	0x01	halt on a gross applications level error that is detected in the network code
DUMP	0x02	Works in conjunction with other options: **BUGHALT** - Dump all arriving packets **PROTERR** - Dump header for level of error **NETTRACE**, etc. - Dump headers at trace level
INFOMSG	0x04	Print general informational messages
NETERR	0x08	Display net interface error messages
PROTERR	0x10	Display protocol level error messages
NETRACE	0x20	Trace packet in link level net layer
TMO	0x40	Print message on timeout
APTRACE	0x80	Trace packet through application
TPTRACE	0x100	Transport protocol (UDP/TCP/RVD) trace
IPTRACE	0x200	Trace packet in internet layer
UPCTRACE	0x400	Trace upcall progress - DANGEROUS!

SEE ALSO

upcall - Debug tracing will not occur on packets being processed in ISR context unless both these options are enabled.

COMMAND

dtrap - try to hook debugger

PARAMETERS

None

EXAMPLE

```
INET> dtrap
???
INET> _
```

DESCRIPTION

dtrap causes a call to the porting engineer provided **dtrap()** function to occur.

On the DOS implementation, **dtrap** executes an **int 3** opcode. This will cause most Intel x86 based systems to turn control over to a debugger (such as DOS debug or Codeview), if one is resident. This will also work under some circumstances with ICEs. On a system without a debugger it will have no effect. The method of resuming execution will vary from debugger to debugger.

COMMAND

dump - hexdump incoming packets

PARAMETERS

None

EXAMPLE

```
INET> dump
Packet hex dumping enabled
INET>
INET> host 10.1
active host number set to 10.0.0.1
INET> dump
Packet hex dumping enabled
INET> dump
Packet hex dumping disabled
INET> _
```

DESCRIPTION

This dumps the SNMP protocol portion of all packets received and sent. It ignores other protocols such as ARP and PING.

KNOWN BUGS

With some Ethernet drivers this can hang the system after a received packet is dumped. It is reliable when used on the loopback driver.

COMMAND

linkstats - display link layer specific statistics

PARAMETERS

An optional interface number

EXAMPLE

```
INET> linkstats 1
Packet driver for LCIPKT, ver: 1, ext: 2
Packet driver link stats:
 packets: In:          3          Out:          3
 bytes  : In:       3060          Out:        3060
 errors : In:          0          Out:          0
 pkts dropped:         0      upcalls:          6
pktdrvq: head:00 tail:00 len:0 max:0 min:0
INET> _
```

DESCRIPTION

This command displays statistics for the hardware associated with the given interface. If no interface is specified, 1 is used. The format, content, and accuracy of these statistics will vary from link driver to link driver. This command differs from the **iface** command in that these counters are generally read from the hardware driver, (Packet driver or ODI) and are maintained since the driver was installed, not since the station started as is the case with the iface counters. Exceptions to this principle are the **upcalls**, **pkts** dropped, and **pktdrvq** display above. These are maintained by the lowest levels of the station software and are included here because they are peculiar to the Packet Driver implementation.

In general, packet counts should be obtained from the **iface** command, since those counters are well defined (by MIB-2) and are uniform across all devices.

SEE ALSO

iface

COMMAND

allocsize - set size for alloc() breakpoint

PARAMETERS

Number of bytes to be allocated

EXAMPLE

```
INET> allocsize 128
malloc trap size set to 128
INET> _
```

DESCRIPTION

The stack code expects the porting engineer to provide an implementation of a function called **npalloc()** that the stack uses for memory allocation. **npalloc()** take a single parameter which specifies the number of bytes to be allocated by the call. The code we ship provides implementations of **npalloc()** for many of our supported target systems (16 bit DOS, net186, the VRTX ports, etc).

In these implementations, there is an option to have the code break into a debugger if the value of the passed parameter equals a particular value. The value that causes the code to trap is specified by the **allocsize** command. It can be useful during debugging for finding places in the code where bogus memory allocations are occurring.

COMMAND

upcall - trace received packets

PARAMETERS

None

EXAMPLE

```
INET> upcall
Upcall debugging enabled
INET> _
```

DESCRIPTION

Enables protocol stack trace reporting on incoming packets. **upcall** toggles the **UPCTRACE** bit in the tracing bit mask that is affected by the **debug** command. Its just a shorthand way of setting this one particular bit.

SEE ALSO

debug

4.1 **Statistics** - define NET_STATS in ipport.h_h

COMMAND

arps - display ARP statistics and table

PARAMETERS

None

EXAMPLE

```
INET> ho 10.0.0.80
active host number set to 192.9.200.3
INET> p
ping 0 to 10.0.0.80: Arping for host...
got ping reply; len :62 seq 0 from 192.9.200.3
ping complete; send 1, received 1
INET> arp
arp Requests In: 75,   out: 28
arp Replys   In: 8,   out: 75
X)  MAC Address   iface pend    IP      ctime    ltime
7)  0080C8-E2AF2A   1   N   10.0.0.80   4669168  4670448
INET> _
```

DESCRIPTION

The ARP command displays some **arp** statistics and dumps data from the entries in the current ARP table. The most interesting fields are the MAC address and the associated IP address. the entry "**X)**" field is the position in the ARP table of this entry. Entries are allocated last first, so this compile has 8 ARP entries (0-7). **iface** is the logical interface the **arp** was resolved on. **pend** indicates whether we have an outgoing packet awaiting the results of an **arp** reply. It will almost always be N for no. The time is the internal timestamp when the ARP reply was last referenced. This is kept so that if we run out of room in the ARP table, we can delete the entry that has not been used for the longest time.

This command is included in the menu list if **NET_STATS** was defined in **ipport.h_h**.

COMMAND

ipstat - display IP layer statistics

PARAMETERS

None

EXAMPLE

```
INET> ipstat
IP MIB statistics:
Gateway: NO     default TTL: 30
rcv:  total: 39002    header err: 0    address err: 0
rcv:  unknown Protocols: 0    delivered: 39002
send: total: 2360    discarded: 0     No routes: 0
Routing; forwarded: 0    discarded: 0
Recvd fragments: 0, Frames reassembled: 0
Pkts fragmented: 0, Fragments sent: 0, dropped: 0
Reasm.Timeouts: 0,  Reasm.Errors: 0
INET> _
```

DESCRIPTION

ipstat displays the standard IP SNMP MIB statistics.

This command is included in the menu list if **NET_STATS** was defined in **ipport.h_h**.

COMMAND

icmpstat - display ICMP layer statistics

PARAMETERS

None

EXAMPLE

```
INET> icmpstat
ICMP layer stats:
icmpInMsgs 1    icmpInErrors 0
In counters: DestUnreach 0   TimeExceed 0   ParmProb 0
SrcQuench 0   Redirect 0   Echo(ping) 0   EchoReps 1
Timestmp  0   TStmpRep 0    AddrMasks 0   AddrMaskRep 0
icmpOutMsgs 2    icmpOutErrors 0
Out counts:  DestUnreach 2   TimeExceed 0   ParmProb 0
SrcQuench 0   Redirect 0   Echo(ping) 0   EchoReps 0
Timestmp  0   TStmpRep 0    AddrMasks 0   AddrMaskRep 0
INET> _
```

DESCRIPTION

icmpstat displays the standard ICMP SNMP MIB statistics

This command is included in the menu list if **NET_STATS** was defined in **ipport.h_h**.

COMMAND

udp - display UDP layer statistics

PARAMETERS

None

EXAMPLE

```
INET> udp
UDP MIB dump:
In: Good: 26    No Port: 2     Bad: 0
Out: 26
INET> _
```

DESCRIPTION

The **udp** command displays the standard UDP SNMP MIB statistics.

This command is included in the menu list if **NET_STATS** was defined in **ipport.h_h**.

4.2 **DNS** - define DNS_CLIENT and NET_STATS in ipport.h_h

COMMAND

dns_stats - display DNS setup and statistics

PARAMETERS

None

EXAMPLE

```
INET> dns_stats
DNS servers:0.0.0.0 0.0.0.0 0.0.0.0 DNS cache:
name: , IP: 0.0.0.0, retry:0, ID:0, rcode:0, err:0
name: , IP: 0.0.0.0, retry:0, ID:0, rcode:0, err:0
name: , IP: 0.0.0.0, retry:0, ID:0, rcode:0, err:0
name: , IP: 0.0.0.0, retry:0, ID:0, rcode:0, err:0
name: , IP: 0.0.0.0, retry:0, ID:0, rcode:0, err:0
name: , IP: 0.0.0.0, retry:0, ID:0, rcode:0, err:0
protocol/implementation runtime errors:0
requests sent:0
replys received:0
usable replys:0
total reties:0
timeouts:0
INET> _
```

DESCRIPTION

This command shows statistics about the DNS Client. It shows a list of DNS servers that the client is configured to know about, a listing of the client side DNS cache which is a kind of a history of DNS resolutions that the client has performed, and some general DNS statistics.

This command is included in the menu list if **DNS_CLIENT** and **NET_STATS** was defined in **ipport.h_h**.

4.3 **TCP** - define INCLUDE_TCP and NET_STATS in ipport.h_h

COMMAND

mbuf - display mbuf statistics

PARAMETERS

None

EXAMPLE

```
INET> mbuf
mfreeq: head:6F79:D9C6, tail:6F79:D5FA, len:59, min:3, max:60
mbufq: head:6F79:D402, tail:6F79:D402, len:1, min:0, max:57
mbuf allocs: 23538, frees: 23536
INET> _
```

DESCRIPTION

BSD implementation of TCP uses **mbuf** for dynamic memory requirements.

This command displays information about the queues that are used to hold freed and in-use mbufs; whereas the **mlist** command below displays detailed information about each mbuf that is in use.

This command is included in the menu list if **INCLUDE_TCP** and **NET_STATS** was defined in **ipport.h_h**.

COMMAND

mlist - display mbufs in use

PARAMETERS

None

EXAMPLE

```
INET> mlist
mbufs in use:
type 2, pkt:FFFFC0D2, data:FFFFC13E, len:18
INET> _
```

DESCRIPTION

This command is included in the menu list if **INCLUDE_TCP** and **NET_STATS** was defined in **ipport.h_h**.

COMMAND

tcp - display TCP statistics

PARAMETERS

None

EXAMPLE

```
INET> tcp
tcpRtoAlgorithm 0,   tcpRtoMin 0
tcpRtoMax 0,         tcpMaxConn 0
tcpActiveOpens 11,   tcpPassiveOpens 6
tcpAttemptFails 0,   tcpEstabResets 3
tcpCurrEstab 0,      tcpInSegs 2344
tcpOutSegs 2567,     tcpRetransSegs 16
tcpInErrs 0,         tcpOutRsts 0
INET> _
```

DESCRIPTION

Displays the standard TCP SNMP MIB statistics.

This command is included in the menu list if **INCLUDE_TCP** and **NET_STATS** was defined in **ipport.h_h**.

COMMAND

sockets - display socket list

PARAMETERS

None

EXAMPLE

```
INET> sockets
TCP sock, fhost,     ports,    opts, rxbytes,state:
0000155A,  10.0.0.80, 23->2908, 0x0100, 0, ESTABLISHED
FFFFDF3E,  0.0.0.0, 23->0, 0x0102, 0, LISTEN
FFFFDE2C,  0.0.0.0, 7->0, 0x0102, 0, LISTEN
FFFFDC88,  0.0.0.0, 21->0, 0x0102, 0, LISTEN
FFFFDB76,  0.0.0.0, 80->0, 0x0102, 0, LISTEN
INET> _
```

DESCRIPTION

Display the list of open sockets.

This command is included in the menu list if **INCLUDE_TCP** and **NET_STATS** was defined in **ipport.h_h**.

COMMAND

tbconn - TCP BSD connection statistics

PARAMETERS

None

EXAMPLE

```
INET> tbconn
connections initiated: 11,      connections accepted: 6
connections established: 12,    connections dropped: 3
embryonic connections dropped: 5, conn. closed (includes drops): 22
segs where we tried to get rtt: 1382,   times we succeeded: 1369
delayed acks sent: 0,   conn. dropped in rxmt timeout: 0
retransmit timeouts: 16,        persist timeouts: 0
keepalive timeouts: 101,        keepalive probes sent: 0
connections dropped in keepalive: 5
INET> _
```

DESCRIPTION

Show the statistics about TCP connections.

This command is included in the menu list if **INCLUDE_TCP** and **NET_STATS** was defined in **ipport.h_h**.

COMMAND

tbsend - TCP BSD send statistics

PARAMETERS

None

EXAMPLE

```
INET> tbsend
total packets sent: 2612,       data packets sent: 1366
data bytes sent: 68491, data packets retransmitted: 1
data bytes retransmitted: 3,    ack-only packets sent: 1211
window probes sent: 0,  packets sent with URG only: 0
window update-only packets sent:0, control (SYN|FIN|RST) packets
sent:34
INET> _
```

DESCRIPTION

Show statistics about TCP packets sent (for various connections).

This command is included in the menu list if **INCLUDE_TCP** and **NET_STATS** was defined in **ipport.h_h**.

COMMAND

tbrcv - TCP BSD receive statistics

PARAMETERS

None

EXAMPLE

```
INET> tbrcv
total packets received: 2401,   packets received in sequence: 1149
bytes received in sequence: 2435,    packets received with ccksum
errs: 0
packets received with bad offset: 0,    packets received too
short: 0
duplicate-only packets received: 62,    duplicate-only bytes
received: 62
packets with some duplicate data: 0,   dup. bytes in part-dup.
packets: 0
out-of-order packets received: 8,       out-of-order bytes
received: 0
packets with data after window: 0,      bytes rcvd after window: 0
packets rcvd after close: 0,    rcvd window probe packets: 0
rcvd duplicate acks: 9, rcvd acks for unsent data: 0
rcvd ack packets: 1384, bytes acked by rcvd acks: 68811
rcvd window update packets: 0
INET> _
```

DESCRIPTION

Show statistics for TCP packets received (for various TCP connections).

This command is included in the menu list if **INCLUDE_TCP** and **NET_STATS** was defined in **ipport.h_h**.

4.4 **Modem** - define USE_MODEM in ipport.h_h

COMMAND

hangup - hang up (and reset) the modem

PARAMETERS

None

EXAMPLE

```
INET> hangup
INET> _
```

DESCRIPTION

Hang up and reset the modem. This will also shut down all protocols (e.g. PPP) running over this interface.

This command is included in the menu list if **USE_MODEM** was defined in **ipport.h_h**.

COMMAND

modem - modem, dialer, and UART info

PARAMETERS

None

EXAMPLE

```
INET> modem
unit 0, dialer state: CONNECTED
last baud rate: 21600
modem DCD(2fe): 0xb0, Lines(2fc): 0xb0
UART com port is com2
UART interrupts: 595
UART rcvd bytes: 595
UART read bytes: 595
UART sent bytes: 1102
UART send timeouts: 0
INET> _
```

DESCRIPTION

Displays various statistics related to modem usage.

This command is included in the menu list if **USE_MODEM** and **NET_STATS** was defined in **ipport.h_h**.

4.5 **HTTP** - define WEBPORT in ipport.h_h

COMMAND

dir - directory of VFS files

PARAMETERS

None

EXAMPLE

```
INET> dir
 name    flags size  (compressed)
btnmap.map     ----  0     (----0)
hlpmask.htm    -0--  694   (----0)
setip.htm      -0--  1320  (----0)
index.htm      -0--  1482  (----0)
body.htm -0--  65    (----0)
inworks.htm    -0--  168   (----0)
helphlp.htm    -0--  588   (----0)
hlpipadd.htm   -0--  1484  (----0)
statmenu.htm   -0--  1154  (----0)
setport.htm    -0--  1699  (----0)
ptstats.htm    -0--  616   (----0)
prtstat.htm    -0--  157   (----0)
pdetail.htm    -0--  1772  (----0)
helpbtn.gif    ----  1197  (----0)
nplogot.gif    ----  1781  (----0)
hub4907.gif    ----  6864  (----0)
btnmap.gif     ----  3943  (----0)
ipaddr   ---s  0     (----0)
ipmask   ---s  0     (----0)
defgw    ---s  0     (----0)
hpport   ---s  0     (----0)
frmsent  ---s  0     (----0)
frmrvcd  ---s  0     (----0)
bytesent ---s  0     (----0)
bytervcd ---s  0     (----0)
ptcolls  ---s  0     (----0)
pterrors ---s  0     (----0)
portstat ---s  0     (----0)
setip.cgi      --x-  0     (----0)
setport.cgi    --x-  0     (----0)
24984 bytes in 30 vfs files
INET> _
```

DESCRIPTION

This command is included in the menu list if **WEBPORT** was defined in **ipport.h_h**.

COMMAND

hstat - HTTP statistics

PARAMETERS

None

EXAMPLE

```
INET> hstat
HTTP stats: requests:119 gets:118 errors:0, ssi:117 cgi:0
No http connections currently open.
INET> _
```

DESCRIPTION

This command is included in the menu list if **WEBPORT** was defined in **ipport.h_h**.

4.6 **PPP** - define USE_PPP in ipport.h_h

COMMAND

pfile - log PPP events to the file ppp.log

PARAMETERS

None

EXAMPLE

```
INET> pfile
ppp file logging turned ON
INET> pfile
ppp file logging turned OFF
INET> _
```

DESCRIPTION

pfile toggles the state of whether PPP trace information will be logged to a file or not.

This command is included in the menu list if **USE_PPP** was defined in **ipport.h_h**.

COMMAND

pcons - log PPP events to console

PARAMETERS

None

EXAMPLE

```
INET> pcons
ppp file logging turned ON
INET> pcons
ppp file logging turned OFF
INET> _
```

DESCRIPTION

pcons toggles the state of whether PPP trace information will be logged to the target system console or not.

This command is included in the menu list if **USE_PPP** was defined in **ipport.h_h**.

COMMAND

iface - display net interface statistics

PARAMETERS

Optional interface number

EXAMPLE

```
INET> iface
Interface (MIB 'if' group) stats for Interface 1, Ethernet Packet
Driver
rcvd: errors:0   dropped:0   station:0   bcast:0   bytes:0
sent: errors:0   dropped:0   station:0   bcast:0   bytes:0
INET> iface 2
Interface (MIB 'if' group) stats for Interface 2, Ethernet ODI
Driver
rcvd: errors:0   dropped:0   station:16   bcast:0   bytes:735
sent: errors:0   dropped:0   station:7   bcast:1   bytes:611
INET> iface 3
Interface (MIB 'if' group) stats for Interface 3, loopback
rcvd: errors:0   dropped:0   station:0   bcast:0   bytes:0
sent: errors:0   dropped:0   station:0   bcast:0   bytes:0
INET> _
```

DESCRIPTION

The **iface** command displays statistics for the given interface. If no interface is specified, the default is 1. The statistics displayed in the command are a subset of those defined for interfaces in MIB-2 (RFC1213). The numbering of the interfaces also corresponds to the interface indexing described in MIB-2. This indexing scheme starts numbering at one (i.e. three interfaces would be numbered 1, 2, 3), whereas the InterNiche protocol stack's internal mechanisms maintain the interface indexes numbered from 0 (i.e. 0, 1, 2). This should be kept in mind by programmer's working on the InterNiche source code.

This command is included in the menu list if **NET_STATS** was defined in **ipport.h_h**.

SEE ALSO

linkstats, **open**

4.7 **Memory** - define MEM_BLOCKS in ipport.h_h

COMMAND

memory - list currently allocated memory

PARAMETERS

None

EXAMPLE

```
INET> memory
block[0] at 4297:037C, 40 bytes
block[1] at 4297:03A6, 1536 bytes
block[2] at 4297:09A8, 40 bytes
block[3] at 4297:09D2, 1536 bytes
block[4] at 4297:0FD4, 40 bytes
block[5] at 4297:0FFE, 1536 bytes
block[6] at 4297:1600, 40 bytes
block[7] at 4297:162A, 1536 bytes
block[8] at 4297:1C2C, 40 bytes
block[9] at 4297:1C56, 1536 bytes
block[10] at 4297:2258, 40 bytes
block[11] at 4297:2282, 128 bytes
block[12] at 4297:2304, 40 bytes
block[13] at 4297:232E, 128 bytes
block[14] at 4297:23B0, 40 bytes
block[15] at 4297:23DA, 128 bytes

... etc.

INET> _
```

DESCRIPTION

Dumps address and size of all allocated but un-freed memory since the application started. This is useful in detecting memory leaks during ports on the InterNiche SNMP API when no other such tools are available.

This command is included in the menu list if **MEM_BLOCKS** was defined in **ipport.h_h**.

4.8 IP - define IP_ROUTING in ipport.h_h

COMMAND

routes - display IP route table

PARAMETERS

None

EXAMPLE

```
INET> routes
..IPaddr.......mask.........nexthop...iface..type
10.0.0.20  255.255.255.255  10.0.0.20  1   OTHER
10.0.0.80  255.255.255.255  10.0.0.80  1   OTHER
10.0.0.22  255.255.255.255  10.0.0.22  1   OTHER
127.0.0.1  255.255.255.255  127.0.0.1  2   OTHER
INET> _
```

DESCRIPTION

This command is included in the menu list if **IP_ROUTING** was defined in **ipport.h_h**.

COMMAND

rtadd - manually add IP route to table

PARAMETERS

First three parameters **target.ip**, **target.mask**, and **next.hop** in IP dot notation followed by **0**, **1**, or **2** for the interface number

EXAMPLE

```
INET> rtadd
usage: target.ip target.mask next.hop iface
 where 1st 3 parms are in IP dot notation, last is digit 0-2
INET> rtadd 10.0.0.25  255.255.255.255  10.0.0.25  1
INET> _
```

DESCRIPTION

This command is included in the menu list if **IP_ROUTING** was defined in **ipport.h_h**.

4.9 **SNMP** - define INCLUDE_SNMP in ipport.h_h

COMMAND

snmp - display SNMP MIB counters

PARAMETERS

None

EXAMPLE

```
INET> snmp
snmpInPkts: 0              snmpOutPkts: 0
snmpInBadVersions: 0       snmpInBadCommunityNames: 0
snmpInBadCommunityUses: 0  snmpInASNParseErrs: 0
snmpInTooBigs: 0           snmpInNoSuchNames: 0
snmpInBadValues: 0         snmpInReadOnlys: 0
snmpInGenErrs: 0           snmpInTotalReqVars: 0
snmpInTotalSetVars: 0      snmpInGetRequests: 0
snmpInGetNexts: 0          snmpInSetRequests: 0
snmpInGetResponses: 0      snmpInTraps: 0
snmpOutTooBigs: 0          snmpOutNoSuchNames: 0
snmpOutBadValues: 0        snmpOutGenErrs: 0
snmpOutGetRequests: 0      snmpOutGetNexts: 0
snmpOutSetRequests: 0      snmpOutGetResponses: 0
snmpOutTraps: 0            snmpEnableAuthenTraps: 2
INET> _
```

DESCRIPTION

Displays the counters associated with the SNMP protocol layer. These counters are those described for SNMP in MIB-2 (RFC1213).

This command is included in the menu list if **INCLUDE_SNMP** was defined in **ipport.h_h**.

5. PROTOCOL SPECIFIC COMMANDS

5.1 DHCP Server

FILE

dhcpmenu.c

COMMAND

help dhcpsrv - displays a list of DHCP server commands

PARAMETERS

None

EXAMPLE

```
INET> ? dhcpsrv
   dhsrv       DHCP server statistics
   dhlist      DHCP server assigned addresses
   dhentry     list specific entry details
   dhdelete    delete a DHCP entry
INET> _
```

DESCRIPTION

Shows the command set for DHCP Server.

COMMAND

dhsrv - DHCP server statistics

PARAMETERS

None

EXAMPLE

```
INET> dhsrv
plain bootp requests received: 0
plain bootp replys sent: 0
discover packets received: 0
offer packets sent: 0
dhcp request packets received: 0
declines received: 0
releases received: 0
acks sent: 0
naks sent: 0
requests for other servers: 0
protocol errors; all types: 0
INET> _
```

DESCRIPTION

Show the statistics for DHCP Server.

COMMAND

dhlist - DHCP server assigned addresses

PARAMETERS

None

EXAMPLE

```
INET> dhlist
no DHCP/BOOTP entrys in database
INET> dhlist
   IP addr     client ID         type      status    lease
1 10.0.0.151 00:40:05:E5:1C:E6 dynamic   Assigned via DHCP 2285
2 10.0.0.34 00:00:60:90:06:8C dbase     unassigned 0
3 10.0.0.33 00:00:F4:90:0E:D8 dbase     unassigned -1

3 Entries
INET> _
```

DESCRIPTION

Show the list of IP addresses assigned by the DHCP Server.

COMMAND

dhentry - list specific entry details

PARAMETERS

Index number

EXAMPLE

```
INET> dhentry
Need a dh_entry index number (1 - ?).
dhcp entry index out of range - try one of these:
no DHCP/BOOTP entrys in database
INET> dhentry
Need a dh_entry index number (1 - ?).
dhcp entry index out of range - try one of these:
   IP addr     client ID         type      status    lease
1 10.0.0.151 00:40:05:E5:1C:E6 dynamic   Assigned via DHCP 3331
2 10.0.0.34 00:00:60:90:06:8C dbase     unassigned 0
3 10.0.0.33 00:00:F4:90:0E:D8 dbase     unassigned -1

3 Entries
INET> _
```

DESCRIPTION

Show the details about a particular entry. For each IP address that is assigned by the DHCP Server, an entry is made in the DHCP database.

COMMAND

dhdelete - delete a DHCP entry

PARAMETERS

Index number

EXAMPLE

```
INET> dhdelete
Need a dh_entry index number (1 - ?).
INET> dhlist
   IP addr     client ID         type      status    lease
1 10.0.0.151 00:40:05:E5:1C:E6 dynamic   Assigned via DHCP 2285
2 10.0.0.34 00:00:60:90:06:8C dbase     unassigned 0
3 10.0.0.33 00:00:F4:90:0E:D8 dbase     unassigned -1

3 Entries
INET> dhdelete 2
deleted
INET> dhlist
   IP addr     client ID         type      status    lease
1 10.0.0.151 00:40:05:E5:1C:E6 dynamic   Assigned via DHCP 2082
2 10.0.0.33 00:00:F4:90:0E:D8 dbase     unassigned -1

2 Entries
INET> _
```

DESCRIPTION

Use the **dhlist** command to get information about a DHCP entry and then use **dhdelete** command to delete that entry from the DHCP database. Please note that this command only affects the DHCP database. DHCP Server has no means to prevent the remote device/computer from using the IP address that would be freed by this call.

5.2 Email Alerter

FILE

smtpport.c

COMMAND

help smtp - displays a list of SMTP commands

PARAMETER

smtp

EXAMPLE

```
INET> ? smtp
   mdel        delete an SMTP alert recipient
   mport       TCP port for SMTP alerts
   mrcpt       add/view SMTP alert recipients
   mserver     SMTP server IP address
   mtest       Send a test SMTP alert
   mfile       Email a disk file
   mstat       dump SMTP info
   mverbose    toggle SMTP verbose mode
INET> _
```

DESCRIPTION

Displays the command set for Email Alerter.

COMMAND

mdel - delete an SMTP alert recipient

PARAMETERS

Email address to be deleted, without parameters it lists the current recipients.

EXAMPLE

```
INET> mdel
Specify recipient to delete.
SMTP Alerts Enabled
SMTP Server IP address 207.156.252.7:25, currently disconnected,
state:0.
Alert Recipients:
        jbartas@iniche.com
        llarder@iniche.com
Alert Msgs in Queue: 0
Message fates; OK: 0, bad code: 0, so_error: 0
INET> mdel llarder@iniche.com
recipient deleted
INET> _
```

DESCRIPTION

Email Alerter is informed about email addresses which receive the alerts. This command deletes the email address of a particular recipient. The **mrcpt** command is used to add email addresses to the recipient list.

COMMAND

mport - TCP port for SMTP alerts

PARAMETERS

Optional - new port number.

EXAMPLE

```
INET> mport
smtp server port is currently 25
to change, enter new port number after this command
INET> mport 27
INET> mport
smtp server port is currently 27
to change, enter new port number after this command
INET> _
```

DESCRIPTION

mport shows the port number that is being used for setting up a TCP connection to send the emails. This command can also be used to change the port number.

The port number being specified is the port number on the remote server machine to which the connection attempts are going to be made.

COMMAND

mrcpt - add/view SMTP alert recipients

PARAMETERS

Optional - new email address to be added to recipient list

EXAMPLE

```
INET> mrcpt
SMTP Alerts Enabled
SMTP Server IP address 207.155.248.7:25, currently disconnected,
state:0.
Alert Recipients:
        jbartas@iniche.com
Alert Msgs in Queue: 0
Message fates; OK: 0, bad code: 0, so_error: 0
To add recipient, type email address after command
INET> mrcpt llarder@iniche.com
INET> mrcpt
SMTP Alerts Enabled
SMTP Server IP address 207.155.248.7:25, currently disconnected,
state:0.
Alert Recipients:
        jbartas@iniche.com
        llarder@iniche.com
Alert Msgs in Queue: 0
Message fates; OK: 0, bad code: 0, so_error: 1
To add recipient, type email address after command
INET> _
```

DESCRIPTION

This command is used to view the list of email addresses in the recipient list. Emails will be sent to all recipients in this list. This command can also be used to add new email addresses to the recipient list.

COMMAND

mserver - SMTP server IP address

PARAMETERS

Optional - new server address

EXAMPLE

```
INET> mserver
smtp server is currently 207.155.248.7
to change, enter new IP address after this command
INET> mserver 207.156.252.7
INET> mserver
smtp server is currently 207.156.252.7
to change, enter new IP address after this command
INET> _
```

DESCRIPTION

This command is used to specify the IP address of the SMTP Server. When an email is to be sent, a SMTP connection is made to this server. Without a SMTP server, no emails can be sent.

COMMAND

mtest - send a test SMTP alert

PARAMETERS

None

EXAMPLE

```
INET> mtest
INET> _
```

DESCRIPTION

This command sends a test email to everybody in the recipient list.

Before using this command, the IP address of the SMTP Server should be set and also the recipient list.

COMMAND

mfile - email a disk file

PARAMETERS

Name of the file to be sent

EXAMPLE

```
INET> mfile
INET> mfile webport.nv
INET> _
```

DESCRIPTION

This command sends the specified file as an email attachment to everybody in the recipient list.

Before using this command, the IP address of the SMTP Server should be set and also the recipient list.

COMMAND

mstat - dump SMTP info

PARAMETERS

None

EXAMPLE

```
INET> mstat
SMTP Alerts Enabled
SMTP Server IP address 207.155.248.7:25, currently disconnected,
state:1.
Alert Recipients:
        jbartas@iniche.com
        llarder@iniche.com
Alert Msgs in Queue: 2
msg at 65E8:F930, state: 1
msg at 65E8:FAEC, state: 1
last message reply 0 polls (5267 seconds) ago
Message fates; OK: 0, bad code: 0, so_error: 0
INET> _
```

DESCRIPTION

Show statistics about the Email Alerter module.

COMMAND

mverbose - toggle SMTP verbose mode

PARAMETERS

None

EXAMPLE

```
INET> mverbose
verbose mode ON
INET> mverbose
verbose mode OFF
INET> _
```

DESCRIPTION

When the verbose mode is set, then information is displayed in detail for all the emails sent.

5.3 FTP Client

FILE

ftpmenu.c

COMMAND

help ftpc - displays a list of FTP client commands

PARAMETERS

ftpc

EXAMPLE

```
INET> ? ftp
   ascii       use ASCII transfer mode
   binary      use Binary transfer mode
   cd          change server's directory
   fclose      close FTP command connection
   fverb       toggle verbose mode
   fpasv       set server to passive mode
   ftp         open an FTP connection
   hash        toggle hash mark printing
   get         GET a file
   put         PUT a file
   pwd         print working directory
   ls          list files in server directory
   fstate      display FTP client state
INET> _
```

DESCRIPTION

Displays the list of FTP client commands.

COMMAND

ascii - use ASCII transfer mode

PARAMETERS

None

EXAMPLE

```
INET> ascii
ftp send: TYPE A
INET> ftp reply: 200 Command OK
```

DESCRIPTION

The **ascii** command has the same meaning here as it does in a standard FTP client program. It specifies that files are to be transferred in ASCII, as opposed to binary, form.

COMMAND

binary - use Binary transfer mode

PARAMETERS

None

EXAMPLE

```
INET> binary
ftp send: TYPE I
INET> ftp reply: 200 Command OK
```

DESCRIPTION

The **binary** command has the same meaning here as it does in a standard FTP client program. It specifies that files are to be transferred in binary, as opposed to ASCII, form.

COMMAND

cd - change server's directory

PARAMETERS

None

EXAMPLE

```
INET> cd
please specify path arg
INET> cd subdir
ftp send: CWD subdir
INET> ftp reply: 200 directory changed to c:\subdir
ftp: directory changed to c:\subdir
IINET> cd ..
ftp send: CWD ..
INET> ftp reply: 200 directory changed to c:\
ftp: directory changed to c:\
INET> _
```

DESCRIPTION

Same as the standard FTP **cd** command, this changes the server's current directory.

COMMAND

fclose - close FTP command connection

PARAMETERS

None

EXAMPLE

```
INET> fclose
INET> _
```

DESCRIPTION

Closes the FTP command connection.

COMMAND

fverb - toggle verbose mode

PARAMETERS

None

EXAMPLE

```
INET> fverb
ftp verbose mode off
INET> fverb
ftp verbose mode on
INET> _
```

DESCRIPTION

Toggles the verbose mode on and off.

COMMAND

fpasv - set server to passive mode

PARAMETERS

None

EXAMPLE

```
INET> fpasv
INET> _
```

DESCRIPTION

Sets the FTP server to passive mode.

COMMAND

ftp - open an FTP connection

PARAMETERS

IP address
user name
password

EXAMPLE

```
INET> ftp
usage: ftp host username [password]
INET> ftp 10.0.0.22 guest sesame
INET> ftp reply: 220 Service ready
ftp send: USER guest
ftp reply: 331 User name ok, need password
ftp send: PASS sesame
ftp reply: 230 User logged in
ftp user "guest" logged in.
INET> FTP Connection timed out : No activity for 600 secs
INET> _
```

DESCRIPTION

The **ftp** command takes from 2 to 3 parameters. The first parameter is the IP address of the FTP server to which we are trying to connect. The second parameter is the user name that we want to login to the server as. The third parameter is that user's password. The password is optional. This command will attempt to create an FTP command connection to the specified server. The **fclose** command closes this command connection.

COMMAND

hash - toggle hash mark printing

PARAMETERS

None

EXAMPLE

```
INET> hash
FTP hash mark printing turned on
INET> hash
FTP hash mark printing turned off
INET> _
```

DESCRIPTION

Most FTP clients support this option as a way for the user to monitor the progress of an FTP data transfer. When enabled, the client displays hash, #, marks every so often to let the user know that the data transfer is progressing.

COMMAND

get - GET a file

PARAMETERS

filename

EXAMPLE

```
INET> get readme.txt readme.000
ftp send: PORT 10,0,0,68,31,67
INET> ftp reply: 200 Command OK
ftp send: RETR readme.txt
ftp reply: 150 Here it comes...
receiving file...
INET> ###ftp server closed data connection
INET> ftp reply: 226 Transfer OK, Closing connection
Received 15068 bytes in 0 seconds
INET> _
```

DESCRIPTION

get takes 1 or 2 parameters. The first required parameter is the name of the file on the remote FTP server that is to be retrieved to the local client machine. The second optional parameter specifies the name that should be assigned to the file that is created on the local machine. If the second parameter is not specified, the local file name will be the same as the remote one.

COMMAND

put - PUT a file

PARAMETERS

filename

EXAMPLE

```
INET> put readme.txt
ftp send: PORT 10,0,0,68,31,86
INET> ftp reply: 200 Command OK
ftp send: STOR readme.txt
ftp reply: 150 Connecting for STOR
sending file...
INET> Sent 12495 bytes in 0 seconds
INET> ftp reply: 226 Transfer OK, Closing connection
```

DESCRIPTION

put takes 1 or 2 parameters. The first required parameter is the name of the file on the local client machine that is to be sent to the remote FTP server machine. The second optional parameter specifies the name that should be assigned to the file that is created on the remote machine. If the second parameter is not specified, the remote file name will be the same as the local one.

COMMAND

pwd - print working directory

PARAMETERS

None

EXAMPLE

```
INET> pwd
ftp send: XPWD
INET> ftp reply: 257 "c:\work"
ftp: "c:\work"
INET> _
```

DESCRIPTION

The **pwd** command tells the remote FTP server to print the current working directory that the client is in on the remote machine's file system.

COMMAND

ls - list files in server directory

PARAMETERS

None

EXAMPLE

```
INET> ls
WINDIFF
BUFFER
ADITYA
NEWDEV
NEWBUG
PFE
BUGFIX
MYDOWN~1
VTCP
MATE
CMSOUR~1
MAKESRC
CMSOURCE
T-VTCP~2
OFFICE51
FFASTUNT.FFL
T-VTCP~1
T-VTCP
SLIP
ATULFTP
CONFIG.SYS
ftp server closed data connection
INET> ftp reply: 226 Transfer OK, Closing connection
Listed 81 bytes in 0 seconds
INET> _
```

DESCRIPTION

The **ls** command tells the remote FTP server to list the contents of the current working directory. Its like typing **dir** at a DOS prompt, except its the directory listing of the remote server machine.

COMMAND

fstate - display FTP client state

PARAMETERS

None

EXAMPLE

```
INET> fstate
state: command in progress, mode:ascii
server: 10.0.0.70, data port:20
Hashing: OFF, passive: off
INET> _
```

DESCRIPTION

Displays information about FTP clients.

This command is included in the menu list if **NET_STATS** was defined in **ipport.h_h**.

5.4 Ping

FILE

app_ping.c

COMMAND

help ping - menu to set/view values for ping

PARAMETER

ping

EXAMPLE

```
INET> ? ping
   ping        Ping [host] [#times]
   delay       set milliseconds to wait between pings
   host        set default active IP host
   length      set default ping packet length
   endping     terminate the current ping session
   pstats      display statistics about ping
INET> _
```

DESCRIPTION

Shows the command set for **ping**.

COMMAND

ping - an ordinary ping utility

PARAMETERS

host
#times

EXAMPLE

```
INET> ping 192.9.200.3
ping 0 to 192.9.200.3: Arping for host...
got ping reply; len :62 seq 0 from 192.9.200.3
ping complete; send 1, received 1
INET> host 192.9.200.3
active host number set to 192.9.200.3
INET> ping 3
...press any key to stop pinging...
ping 0 to 192.9.200.3: ping 0 sent...
got ping reply; len :62 seq 0 from 192.9.200.3
ping 1 to 192.9.200.3: ping 1 sent...
got ping reply; len :62 seq 1 from 192.9.200.3
ping 2 to 192.9.200.3: ping 2 sent...
got ping reply; len :62 seq 2 from 192.9.200.3
ping complete; send 3, received 3
INET> _
```

DESCRIPTION

The length and inter packet delay of the **ping**s can be set with the **length** and **delay** commands. **ping** can be sent to any host at any time by specifying the IP address of the host on the command line. If no IP address is specified **ping** will use the default host.

SEE ALSO

length, delay

COMMAND

delay - set milliseconds to wait between pings

PARAMETERS

Number of milliseconds

EXAMPLE

```
INET> delay
current ping delay is 1008
to set, enter number of milliseconds on command line.
INET> delay 100
set inter-ping delay to (approx) 100 ms
INET> delay
current ping delay is 56
to set, enter number of milliseconds on command line.
INET> _
```

DESCRIPTION

Sets the time to wait between pings when the multi-packet ping option is used. **delay** is rounded off to the nearest clock tick (1/18th second approx. 56 milliseconds). Default is one second. Setting the time to less than one tick will cause the pings to be sent with no inter-frame delay at all.

COMMAND

host - set default active IP host

PARAMETERS

IP host name or number

EXAMPLE

```
INET> host 192.9.200.3
active host number set to 192.9.200.3
INET> _
```

DESCRIPTION

This command sets the default host for subsequent commands.. **ping** will default to using the host set with a **host** command. If the host is changed while a session is open, the session is closed and a new session is opened with the new host.

SEE ALSO

ping

COMMAND

length - set default ping packet length

PARAMETERS

A number, usually in the range of 60 - 1514

EXAMPLE

```
INET> length 15
CAUTION: 15 is unusual length
INET> len
default ping length is 15
To change it, put new number on command line
INET> len 1500
INET> length
default ping length is 1500
To change it, put new number on command line
INET> _
```

DESCRIPTION

This command sets the length for ping packets. lengths outside of the range 60-1500 (roughly the range of 10 Mbs Ethernet packets) will generate a cautionary message, but will be used anyway. The exact behavior of pings outside of this range will depend on the MAC drivers and hardware used with the interface.

COMMAND

endping - terminate the current ping session

PARAMETERS

None

EXAMPLE

```
INET> ping 10.20 15
...use endping command to stop pinging...
ping 0 to 10.0.0.20: Arping for host...
INET> got ping reply; len :1022 seq 0 from 10.0.0.20
INET> ping 1 to 10.0.0.20: ping 2 sent...
got ping reply; len :1022 seq 1 from 10.0.0.20
INET> ping 2 to 10.0.0.20: ping 3 sent...
got ping reply; len :1022 seq 2 from 10.0.0.20
INET> ping 3 to 10.0.0.20: ping 4 sent...
got ping reply; len :1022 seq 3 from 10.0.0.20
INET> endping
ping complete; sent 4, received 4
INET> _
```

DESCRIPTION

endping stops the current **ping** session.

COMMAND

pstats - display statistics about ping

PARAMETERS

None

EXAMPLE

```
INET> pstats
Default ping delay time:  1960 ms.
Default ping host: 10.0.0.20
Default ping pkt length: 128 bytes
There are 0 ongoing ping sessions.
INET> _
```

DESCRIPTION

pstats displays statistics about the **ping** settings.

NAT RouterNAT InterNiche menu routines. These are tightly coupled to the InterNiche menuing system in **..\misclib\menu*.***. These routines should be portable to systems using the menus. They are not required for basic NAT functionality.

FILE

natmenu.c

COMMAND

help nat - menu to set/view NAT values

PARAMETER

nat

EXAMPLE

```
INET> ? nat
   natstats    display general NAT statistics,
   natconns    display NAT connection table,
   natentry    NAT connection detail,
   naliases    show alias list,
   nproxies    show proxy list
   nxip        expunge IP address from NAT tables
INET> _
```

DESCRIPTION

Displays the command set for the **NAT Router**.

COMMAND

natstats - display general NAT statistics

PARAMETERS

None

EXAMPLE

```
INET> natstats
local IP: 10.0.0.1 local mask: 255.0.0.0
Internet IP: 209.220.44.220 local mask: 255.255.255.0
timeouts: TCP: 500, UDP: 60
local to inet: pkts:1804, bytes:103876
inet to local: pkts:509, bytes:225773
maxmss: 0, max TCP window: 0
Connections: TCP:4, UDP:0, ICMP:1, created: 496, deleted: 491
Errors: cksum: 0, retries: 478, bad packets: 0
Total IP pkts: 2160, Reserved addresses: 363
ENCAP: rx: 00000000, encap: 00000000, decap: 00000000
  mkfrag: 00000000, rxfrag: 000000000
INET> _
```

DESCRIPTION

Display general statistics of the **NAT Router**.

COMMAND

natconns - display NAT connection table

PARAMETERS

None

EXAMPLE

```
INET> natconns
No open Connections
INET> natconns
TCP: 149.1.1.31:17027 <-> 10.0.0.4:1104
Out_port: 1584, pkts: out 27, in 26, state: 4 encap: 0
TCP: 149.1.1.31:17027 <-> 10.0.0.4:1103
Out_port: 1583, pkts: out 1, in 1, state: 4 encap: 0
TCP: 207.82.70.13:443 <-> 10.0.0.5:3370
Out_port: 1582, pkts: out 28, in 21, state: 4 encap: 0
TCP: 207.82.70.13:443 <-> 10.0.0.5:3368
Out_port: 1580, pkts: out 25, in 20, state: 4 encap: 0
TCP: 207.82.70.13:443 <-> 10.0.0.5:3367
Out_port: 1579, pkts: out 22, in 22, state: 4 encap: 0
TCP: 207.82.70.13:443 <-> 10.0.0.5:3366
Out_port: 1578, pkts: out 26, in 22, state: 4 encap: 0
TCP: 207.82.70.13:443 <-> 10.0.0.5:3365
Out_port: 1577, pkts: out 26, in 24, state: 4 encap: 0
TCP: 207.82.70.13:443 <-> 10.0.0.5:3363
Out_port: 1575, pkts: out 36, in 31, state: 4 encap: 0
TCP: 207.155.252.4:80 <-> 10.0.0.5:3321
Out_port: 1490, pkts: out 7, in 1, state: 4 encap: 0
NET> _
```

DESCRIPTION

Lists statistics for all the open NAT connections.

COMMAND

natentry - NAT connection detail

PARAMETERS

An outside port number

EXAMPLE

```
INET> natentry
enter outside port number of connection on command line
use “natconns” command to get port list
INET> natentry 1525
Foreign IP: 205.188.247.0, Local IP: 10.0.0.5
Ports: outside: 1525, inside: 3336, foreign: 80
outgoing: pkts: 8, bytes: 938
incoming: pkts: 6, bytes: 3689
Type TCP, seconds since use 114
TCP Seq: 381308120, Ack: 2083411835, state: 5
Retrys: Local: 0, Foreign: 0
Bad checksum: Local: 0, Foreign: 0
INET> _
```

DESCRIPTION

Show information about a particular NAT connection.

COMMAND

naliases - show alias list

PARAMETERS

None

EXAMPLE

```
INET> naliases
205.206.207.3 aliased to 10.0.0.52
205.206.207.2 aliased to 214.69.218.2
INET> _
```

DESCRIPTION

Shows the list of NAT aliases. Aliases can be setup in the **natdb.nv** file.

This command is included in the menu list if **NAT_ALIASLIST** was defined in **ipport.h_h**.

COMMAND

nproxies - show proxy list

PARAMETERS

None

EXAMPLE

```
INET> nproxies
TCP port 19 mapped to 10.0.0.52:19
TCP port 21 mapped to 10.0.0.52:21
TCP port 80 mapped to 10.0.0.52:80
TCP port 61 mapped to 10.0.0.52:61
INET> _
```

DESCRIPTION

Shows the list of NAT proxies. Proxies can be setup in the **natdb.nv** file.

This command is included in the menu list if **NAT_PROXYLIST** was defined in **ipport.h_h**.

COMMAND

nxip - expunge IP address from NAT tables

PARAMETERS

An IP address

EXAMPLE

```
INET> nxip
enter IP address to expunge on command line
INET> nxip 10.0.0.75
INET> _
```

DESCRIPTION

Expunge IP address from the NAT tables. System administrator might need to do this if some IP addresses on the local network have been changed.

5.5 RIP

Routing Information Protocol Definitions. This file contains code to support RIP commands from the main menu.

FILE

ripmenu.c

COMMAND

help rip - Routing Information Protocol menu

PARAMETER

rip

EXAMPLE

```
INET> ? rip
SNMP Station: rip commands:
   ripstatistics - display RIP statistics
   riproute - display RIP route table
   ripauth  - display RIP authentication table
   riprefuse - display RIP refuse list
   ripglobals - display RIP global list
   ripaddroute - add a route to route table

INET> _
```

DESCRIPTION

Show the command set for RIP.

New routes can be added using the **route** command or **ripaddroute**. If the new route is for RIP, it is recommended that “ripaddroute” is used, because RIP entries keep more details (metrics, etc) about a route.

COMMAND

ripstatistics - display RIP statistics

PARAMETERS

None

EXAMPLE

```
INET> ripstatistics
Showing statistics gathered for RIP protocol.
Number of version errors= 0
Number of address family errors= 0
Number of packets dropped from a host on the refuse list= 0
Refused due to wrong domain for interface= 0
Authentication failures= 0
Unknown authentication type= 0

Now some statistics about each of RIP versions.
Version Number=0
Packets sent; request 0, response 0, reply 0
Packets received; total 0, request 0, response 0
Number of unknown command pkts received = 0

Version Number=1
Packets sent; request 0, response 0, reply 0
Packets received; total 0, request 0, response 0
Number of unknown command pkts received = 0

Version Number=2
Packets sent; request 0, response 0, reply 0
Packets received; total 0, request 0, response 0
Number of unknown command pkts received = 0

INET> _
```

DESCRIPTION

Displays the RIP statistics.

COMMAND

riproute - display RIP route table

PARAMETERS

None

EXAMPLE

```
INET> riproute
Destination..Gateway......Metric..Mask........MainTimer..SecTimer..If
ace
10.0.0.0 10.0.0.22 1 255.0.0.0 1 0 1
Number of routes = 1
INET> ripaddroute 192.9.200.54,255.0.0.0,10.0.0.1,0,1,180,0,0.0.0.0
INET> riproute
Destination..Gateway......Metric..Mask........MainTimer..SecTimer..If
ace
10.0.0.0 10.0.0.22 1 255.0.0.0 1 0 1
192.9.200.54 10.0.0.1 1 255.0.0.0 29627 0 0
Number of routes = 2

INET> _
```

DESCRIPTION

Displays the RIP route table.

COMMAND

ripauth - display RIP authentication table

PARAMETERS

None

EXAMPLE

```
INET> ripauth
Number of entries = 0

INET> _
```

DESCRIPTION

Displays the RIP authentication table.

COMMAND

riprefuse - display RIP refuse list

PARAMETERS

None

EXAMPLE

```
INET> riprefuse
Number of entries = 0

INET> _
```

DESCRIPTION

Displays the IP addresses in the RIP “refuse list”.

COMMAND

ripglobals - display RIP globals list

PARAMETERS

None

EXAMPLE

```
INET> ripglobals
Values of global variables of RIP
rip_default_flags           =1
rip_default_ttl             =180
rip_def_bcast_interval      =30
rip_def_deletion_interval   =120
rip_def_trigger_interval    =5
rip_num_of_ifaces           =1
rip_bcast_timer             =24796
rip_trigger_timer           =1
rip_trigger_timer_interval  =2
rip_allow_default_gateways  =0
Interface..RIP Version Flag(Receive,Send)
1  (3,3)
INET> _
```

DESCRIPTION

Display the values of global variables used by RIP.

COMMAND

ripaddroute - add a route to route table

PARAMETERS

dest - destination IP address
subnetmask - subnet mask
gw - gateway
iface - Interface number
metric - number of hops
ttl - time to live for this entry
flags - flags for this entry:
0 by default
1 (**RIP_PRIVATE**) to be used for permanent entries.
proxy - proxy IP address for RIP-2

EXAMPLE

```
INET> ripaddroute
usage:ripaddroute dest,subnetmask,gw,iface,metric,ttl,flags,proxy
Eg:ripaddroute
192.9.200.54,255.255.255.0,10.0.0.1,1,1,180,0,0.0.0.0
INET> ripaddroute
192.9.200.54,255.0.0.0,10.0.0.1,0,1,180,0,0.0.0.0
INET> riproute
Destination..Gateway......Metric..Mask........MainTimer..SecTimer.
.Iface
10.0.0.0 10.0.0.22 1 255.0.0.0 1 0 1
192.9.200.54 10.0.0.1 1 255.0.0.0 29627 0 0
Number of routes = 2

INET> _
```

DESCRIPTION

Add a route to the RIP table. First, type **ripaddroute** without any arguments to get information about the arguments. This command is tedious, so it always helps to do it this way.

5.6 TELNET

FILE

telmenu.c

COMMAND

help telnet - TELNET server menu

PARAMETER

telnet

EXAMPLE

```
INET> ? telnet
   tshow show the options values for all sessions
   tstats      show the statistics of all TELNET sessions
   logout      logout of the TELNET session
   exit        logout of the TELNET session
INET> _
```

DESCRIPTION

Show the commands for TELNET Server.

COMMAND

tshow - show the options values for all sessions

PARAMETERS

None

EXAMPLE

```
INET> tshow
Showing OPTION values for each telnet session....

Session 1 : Socket is 1870265322......
[0]: configurable=1
   For Local Session:value=0, req sent=0, nego=0
   For Remote Session:value=0, req sent=0, nego=0
[1]: configurable=1
   For Local Session:value=1, req sent=0, nego=1
   For Remote Session:value=1, req sent=0, nego=1
[3]: configurable=1
   For Local Session:value=1, req sent=0, nego=1
   For Remote Session:value=0, req sent=0, nego=1
[5]: configurable=1
   For Local Session:value=0, req sent=0, nego=1
   For Remote Session:value=0, req sent=0, nego=1

Session 2 : Socket is 1870263406......
[0]: configurable=1
   For Local Session:value=0, req sent=0, nego=0
   For Remote Session:value=0, req sent=0, nego=0
[1]: configurable=1
   For Local Session:value=1, req sent=0, nego=1
   For Remote Session:value=1, req sent=0, nego=1
[3]: configurable=1
   For Local Session:value=1, req sent=0, nego=1
   For Remote Session:value=0, req sent=0, nego=1
[5]: configurable=1
   For Local Session:value=0, req sent=0, nego=1
   For Remote Session:value=0, req sent=0, nego=1
INET> _
```

DESCRIPTION

Each TELNET connection negotiates various options to be used. This command shows the values for options used by each TELNET connection.

COMMAND

tstats - show the statistics of all TELNET sessions

PARAMETERS

None

EXAMPLE

```
INET> tstats
Total connections opened = 2
Total connections closed = 0

Telnet Session 1: Showing statistics for socket 1870265322.
Bytes rcvd=103, Cmds rcvd = 12

Telnet Session 2: Showing statistics for socket 1870263406.
Bytes rcvd=47, Cmds rcvd = 3
Number of ongoing telnet sessions = 2.
INET> _
```

DESCRIPTION

Show the statistics for all TELNET sessions.

COMMAND

logout - logout of the TELNET session

PARAMETERS

None

EXAMPLE

```
 logout
INET> _
```

DESCRIPTION

Logout of the TELNET session.

COMMAND

exit - logout of the TELNET session

PARAMETERS

None

EXAMPLE

```
 exit
INET> _
```

DESCRIPTION

Logout of the TELNET session.

6. TARGET SPECIFIC COMMANDS

6.1 Net186

cmain() substitute for **main()**, for AMD Net186 IP router demo.

FILE

cmain.c

COMMAND

help net186 - displays a list of Net186 commands

PARAMETER

net186

EXAMPLE

```
INET> ? net186
SNMP Station: net186 commands:
   uart     - display data uart stats
   usetting - display uart control struct
   uinit    - (re)initialize UART
   baud     - get/set modem UART's baud rate
   telnum   - show/set telephone dial info
   user     - show/set dial-in user name
   pass     - show/set dial-in password
   heaps    - heap (memory) usage statistics

INET> _
```

DESCRIPTION

Displays a list of Net186 diagnostic and (re)configuration commands.

COMMAND

uart - display UART statistics

PARAMETERS

None

EXAMPLE

```
INET> uart
bytes In: 0, Out: 0
unit: 0, interrupts: 0, int_tx: 0
txblock: 0, breaks:0
baud: 0
InputPutPtr: 0,  InputGetPtr: 0
OutputPutPtr: 0, OutputGetPtr: 0
Error mask: 00
Regs; Control: 0xE1, Status: 0x44
Errs: rx overflow: 0, parity: 0, frame: 0
INET> _
```

DESCRIPTION

This command is included in the menu list if **AMD_NET186** was defined in **ipport.h_h**.

COMMAND

usetting - display UART control structure

PARAMETERS

None

EXAMPLE

```
INET> usetting
CtlAddr: FF10
StatAddr: FF12
TxAddr: FF14
RxAddr: FF16
BaudAddr: FF18
IMaskAddr: FF42
EOIAddr: FF22
IntType: 11
IntsEnabled: 10
IntsDisabled: 18
DisabledCtlMask: 41
NoIntCtlMask: 61
RxCtlMask: E1
TxCtlMask: 0100
RxStatMask: 80
TxStatMask: 40
BreakStatMask: 0400
ErrStatMask: 0438
BaudMConst: 10
BaudSConst: 00
PIOEnable: E73F
RxPIOBits: 1080
INET> _
```

DESCRIPTION

usetting displays the contents of a structure that is used to access the board's network UART device.

This command is included in the menu list if **AMD_NET186** was defined in **ipport.h_h**.

COMMAND

uinit - (re)initialize UART

PARAMETERS

None

EXAMPLE

```
INET> uinit
UART enabled, int 0x11, base 0xFF10
reset UART
INET> _
```

DESCRIPTION

uinit initializes the board's network UART device.

This command is included in the menu list if **AMD_NET186** was defined in **ipport.h_h**.

COMMAND

baud - get/set modem UART's baud rate

PARAMETERS

baud rater

EXAMPLE

```
INET> baud
modem UART baud is 19200, enter new speed to change it.
INET> baud 56000
UART enabled, int 0x11, base 0xFF10
set UART baud to 56000
INET> baud
modem UART baud is 56000, enter new speed to change it.
INET> _
```

DESCRIPTION

When this command is invoked without a parameter, it displays the current baud rate. If a parameter is specified, the baud rate is set equal to the parameter.

COMMAND

telnum - show/set telephone dial info

PARAMETERS

phone number

EXAMPLE

```
INET> telnum
current dialout number is 4900610
INET> tel 5551212
dialout set to 5551212
INET> tel
current dialout number is 5551212
INET> _
```

DESCRIPTION

When this command is invoked without a parameter, it displays the current dial out phone number. If a parameter is specified, the dial out phone number is set equal to the parameter.

COMMAND

user - show/set dial-in user name

PARAMETERS

user name

EXAMPLE

```
INET> user
current dial-in user name is ch/iniche.com
INET> user surely
dial-in user name set to surely
INET> user
current dial-in user name is surely
INET> _
```

DESCRIPTION

When no parameter is specified, the current ISP dialin/login user name is displayed. When a parameter is specified, the user name is set to the specified parameter.

COMMAND

pass - show/set dial-in password

PARAMETERS

password

EXAMPLE

```
INET> pass
current dialin password is 07336
INET> pass sesame
dial-in password set to sesame
INET> pass
current dialin password is sesame
INET> _
```

DESCRIPTION

When no parameter is specified, the current ISP dialin/login password is displayed. When a parameter is specified, the password is set to the specified parameter.

COMMAND

heaps - heap (memory) usage statistics

PARAMETERS

None

EXAMPLE

```
INET> heaps
free list:
0: at 6000:014C, 0x16 bytes
1: at 6000:E0E4, 0x1F0C bytes
2: at 6FFF:0000, 0x500A bytes
mh_startfree: 86010
mh_totfree:   28492
mh_minfree:   28326
mh_allocs:    307
mh_fails:     0
mh_frees:     96
INET> _
```

DESCRIPTION

heaps displays information about the memory allocation system that is peculiar to the Net186 board.

Index