Nortel Communication Server 1000

# Communication Server 1000 Release 5.x Troubleshooting Guide for Distributors

Release: 5.x
Document Revision: 01.01

www.nortel.com

NN43001-730

Nortel Communication Server 1000
Release: 5.x
Publication: NN43001-730
Document status: Standard
Document release date: 11 June 2008

# Contents

# About this document

This document contains commands and techniques you can use to troubleshoot problems with Communication Server 1000 (CS 1000) and VoIP components. The information is for administrators and installers who are familiar with the information in the relevant NTPs. This document is intended to take you to the next debug level and provide additional information for those skilled in working directly with the Call Server or the Wind River VxWorks Shell on various platforms.

The commands descriptions in this guide include syntax, examples, and tips to use and interpret results. Nortel recommends that you proceed carefully when you use the commands outlined in this guide.

You require a clear understanding of the architectural differences between CS 1000 and VoIP systems and the features supported for a system.

## Reference documents

- *Security Management Fundamentals,* (NN43001-604)
- *Access Control Management Reference* , (NN43001-602)
- *Enterprise Common Manager Fundamentals,* (NN43001-116)
- *Element Manager System Reference — Administration* , (NN43001-632)
- *IP Phones Fundamentals* , (NN43001-368)
- *Secure Multimedia Controller Fundamentals,* (NN43001-325)

# Introduction

The Communication Server 1000 (CS 1000) and VoIP applications run on a variety of platforms and offer various features depending on the platform:

- ITG-P, SMC, MC32S, MGC, Signaling Server: the five platforms on which the Voice Media Gateway Card (VGMC) application runs. A different binary file exists for each card because each card has a different CPU. The same source code files are used to build all three binary files; therefore, application functionality is the same on all platforms unless system issues prevent it.
- The SMC card has 8 or 32 gateway ports. The ITG-P card has 24 gateway channels. The MC32S has 32 gateway ports. Throughout the document, whenever the channel number is a CLI command parameter, the value range is 0 to 7 or 0 to 31 for the SMC, 0 to 23 for the ITG-P, and 0 to 31 for the MC32S and MGC. The DSP software generates TCID in some printouts; TCID and the channel number are numerically equivalent.
- The term VGMC describes the functionality of the ITG-P, SMC and MC32S cards. In general, these cards are typically gateway cards and the system uses DSP resources as gateways between the TDM and packet domains. Using VGMC means the card can be an ITG-P, an SMC, or an MC32S card. If a feature is specific to one card, only the name of the affected card is used.
- The VGMC application has two main functionalities: the Terminal Proxy Server (TPS) and the voice gateway. The TPS functionality can be used on the Signaling Server or the VGMCs while the voice Gateway function is functional only on the VGMCs and the MGC card.
- The VGMC application is configured on the CS 1000 systems by using the Element Manager (EM) Web tool.

  ***Note:*** CS 1000 Release 5.5 does not support the ITG-P cards. The ITG-P cards are supported only for CS 1000 Release 5.0 and prior releases. All information in this document relating to ITG-P cards is provided for legacy equipment.

## Network sniffer

For network-related problems, a network sniffer helps to see which packets are sent and received by the VGMC/Signaling Server on the TLAN and ELAN interfaces. In many cases, using a sniffer is the only way to determine the cause of an IP Phone registration, voice QoS, or other network messaging problem. Nortel employees who support the IP Line product need access to a sniffer and must understand its use.

Stand-alone sniffers exist but the most cost-effective are those that are a PC application. These applications monitor the subnet to which the PC is connected and display information about the network traffic. The applications can usually monitor only specific port numbers or protocols and display and decode most packet contents.

Nortel recommends the following programs:

- Wireshark (freeware). Go to www.wireshark.org and search for your required platform. Wireshark is a sniffer and protocol analyzer that supports encryption. A decoder for the UNIStim protocol is available for Wireshark. You can obtain the decoder from the FS or GNTS group. Nortel strongly recommends that you install this decoder to help you debug signaling problems.
- Sniffer Pro. You must purchase a separate VoIP decode option with this version. (http://www.sniffer.com) You must have a licence key to run Sniffer Pro. An add-on UNIStim/RUDP message decode package is available for Sniffer Pro. When the decode package installed, the proprietary UNIStim/RUDP messaging is decoded. Nortel strongly recommends that you install this add-on.

Connect a sniffer to the ELAN when you debug VGMC/MGC/Signaling Server to Call Server or VGMC/MGC/Signaling Server to EM problems. Connect the sniffer to the TLAN or the LAN with the IP Phone to debug TPS to IP Phone problems and voice Gateway-related issues. For more information about using a sniffer, see “Network sniffer” (page 27).

## VGMC and Signaling Server VxWorks shell access

The following commands control aspects of the VGMC shell access. While some commands in this document are accepted at the VGMC> shell prompt, all are accepted at the VxWorks shell prompt (->). Use the command **`vxWorksShell`** to access that shell interface on the VGMC. Use **`vxshell`** to access the shell interface on the Signaling Server.

> ***Note:*** Nortel recommends that you use a TTY logon to the VGMC through an ELAN Telnet session instead of a direct serial port connection if you log large amounts of data or copy large SYSLOG or OMREPORT files to the console.

## VGMC, Signaling Server, or MGC maintenance port connections

Connect to the VGMC, Signaling Server, or MGC maintenance port using the following methods.

### VGMC or MGC card connection

Connect a serial cable to the backplane adaptor or to the faceplate connector of the MC32S, ITG-P, or SMC card. The card hardware cannot support two devices connected at once; device damage can occur.

Connect to the cards as follows:

- ITG-P: for the backplane octopus cable, connect to the P2 connector
- SMC: for the backplane L adaptor, connect to the RS-232 9-pin connector
- MC32S: for the backplane L adaptor, connect to the RS-232 9-pin connector
- MGC: install in slot 0 (where the SSC was installed in previous releases); the standard SDI cable breaks out the ports; connect to SDI0

### Modem connection

If a modem directly connects to the VGMC or MC32S, you require a null modem adaptor between the card faceplate or the backplane connector and the modem.

The MGC can support a modem connection on SDI0 only; SDI1 and SD2 do not have hardware flow control. A null modem is not required to connect to SDI0.

A null modem is not required to connect to the Signaling Server.

### Terminal configuration

For the VGMC and MC32S, set the terminal device to 9600,8,N,1. Set flow control to None (Hyperterm terminology) or a similar setting. If hardware flow control is enabled, you can see information from the VGMC but the card does not respond to keystrokes; the condition eventually deteriorates to no printing and no response. If this happens, change the flow control setting back to None, close the session, and reopen it. The VGMC should respond.

The default configuration for the MGC card is 9600,8 N,1 with no flow control. Change this configuration in LD 17.

On the Signaling Server, set the terminal device to 19200,8,N,1. Change this configuration by using the **`stty`** command after you log on.

## Shell access

When you log into a card, a shell prompt appears to indicate the active shell. The shell prompt is different between the VGMC and Signaling Servers. Many commands in this document are available only from the lowest level shell, which requires a full logon sequence to access them. The command section indicates at which shell level the command can run.

### VGMC

These cards have two levels: the VGMC application CLI (VGMC>) and the VxWorks shell (->). The following example shows how to go to and from the VGMC shell on the VGMCs.

```
IPL> vxshell
login:  pdt2
password:
Welcome to the VxWorks Shell
WARNING: Data entry errors in this shell can cause loss of
service.  Use itgShell to return to the ITG shell.
value = 52688160 = 0x323f520
-> -> exit
IPL>
```

### Signaling Server

The Signaling Server has no VGMC> prompt. Instead, an Operation And Administration Management shell oam> prompt appears. All commands available in the oam shell are also available at the vxshell level; therefore, most commands run from the vxshell level (->).

> ***Note:*** To access the vxshell, you must log on as pdt2, not admin1 or admin2.

```
login:
pdt2
password:

The software and data stored on this system are the property
of, or licensed to, Nortel Networks and are lawfully
available only to authorized users for approved purposes.
Unauthorized access to any software or data on this system
is strictly prohibited and punishable under appropriate
laws.  If you are not an authorized user then logout
immediately.  This system may be monitored for operational
purposes at any time.

oam>
<< enter 'ctrl pdt' to get to the pdt> shell >>
pdt> vxshell
```

```
->
-> exit
pdt>
```

## MC32S card

The MC32S card is similar to the Signaling Server; no VGMC> prompt exists, and the pdt shell functions as the vxshell. The PDT shell passes commands to the vxWorks shell so most commands run from the PDT shell level (pdt>). Upon connection to the card, you are prompted with the –`login:`|| string after pressing the Enter key. You can enter a valid user name and password to enter the OAM shell, or type **`^PDT`** to enter either PDT shell. Go to PDT from the OAM shell at any time, but once in the PDT shell, you must exit before going to the OAM shell. Because all OAM-level commands are in PDT, this should work correctly. Logging on through ssh takes you directly to PDT with no OAM access. In any shell, if no keyboard activity occurs for a period of time (20 minutes by default) the shell terminates and you return to the —login prompt from a serial connection or disconnects from a remote logon session. The VxWorks shell and the ssh sessions do not timeout.

The following example shows a PDT shell logon on the MC32S card.

```
oam>
<< enter 'ctrl pdt' to get to the pdt> shell >>

PDT login on /pty/pty00.S
Username:  pdt2
Password:

The software and data stored on this system are the property
of, or licensed to, Nortel Networks and are lawfully
available only to authorized users for approved purposes.
Unauthorized access to any software or data on this system
is strictly prohibited and punishable under appropriate
laws.  If you are not an authorized user then logout
immediately.  This system may be monitored for operational
purposes at any time.
SEC0029 Security Warning:  This system contains insecure
passwords, notify your system administrator

Welcome to the MC32S command line.
Software Version:  no label found
Management IP: 47.11.214.85
Primary CS IP Address:  0.0.0.0
```

```
OS Created on:  Date [Apr 19 2007] Time [12:32:21]

pdt>
```

### MGC card

Because the MGC card can be accessed through the Call Server, the MGC card does not support a PDT shell. Instead, use a Local Diagnostic Shell 1 (LDB1), which has functionality similar to the PDT1 shell on the Call Server. The MGC card also supports an advanced LDB2 shell, similar in functionality to the PDT2 shell on the Call Server.

You can access the LDB shells locally through one of the MGC serial ports or remotely through rlogin, Telnet, secure shell, or PPP. When accessing the MGC, you are prompted to provide a user name and password. If you enter the Call Server PDT1 user name and password, enter the LDB1 shell. If you enter the Call Server PDT2 user name and password, you enter the LDB2 shell. If you enter the LDB1 or LDB2 shell, the command prompt is ldb>. The VxWorks shell can be entered only from the LDB2 shell, by entering **`su`**.

The following example shows an LDB shell logon on the MGC card.

```
oam>
<< enter 'ctrl ldb' to get to the ldb> shell >>

LDB login on /pty/pty00.S
Username:  pdt2
Password:


The software and data stored on this system are the property
of, or licensed to, Nortel Networks and are lawfully
available only to authorized users for approved purposes.
Unauthorized access to any software or data on this system
is strictly prohibited and punishable under appropriate
laws.  If you are not an authorized user then logout
immediately.  This system may be monitored for operational
purposes at any time.
SEC0029 Security Warning:  This system contains insecure
passwords, notify your system administrator

Welcome to the Media Gateway Controller command line.
Firmware Version:  MGCCAD15
Management IP: 47.11.214.83

IPMG: 20 0
Primary CS IP Address:  47.11.214.87
```

```
Installed Daughterboards:  1
OS Created on:  Date [Apr 13 2007] Time [15:24:46]

ldb> su
->
-> exit
ldb>
```

The appropriate Nortel development and Global Customer Care Service staff have the logon information; it is not included here for security reasons.

### Cannot access VGMC through Telnet

When you use a Telnet session to access a card, you may receive no response to your input. Try entering Ctrl+Q from a TTY session though a serial port. If this does not fix the problem, then the card may be locked. For more information, see “VGMC or Signaling Server locks up” (page 34).

### VGMC or Signaling Server corrupted password

If the NVRAM becomes corrupt or another problem prevents you from logging onto the VGMC shell, you can reset the password. Enter the BIOS as the card boots (enter jkl when prompted). Then enter the command **`shellTi`**, which resets the password to the factory default. You can then log on and change the password.

The Signaling Server does not store the passwords in the NVRAM, so this password reset method does not apply to a corrupted Signaling Server password.

## VGMC and Signaling Server directory structure

The remainder of this document refers to directories that store various files used by the VGMC application and the LTPS. The directories and locations of directories vary between the VGMCs and the Signaling Server. The following sections describe the directory locations for each card type.

### VGMC

The VGMC application uses a number of directories on the VGMC /C: flash drive to store configuration, data and log files:

- LOG: contains the SYSLOG.n files
- OM: contains the OMREPORT.nnn files
- CONFIG: contains the CONFIG.INI, BOOTP.TAB, SECURITY.INI, and TPS.INI configuration files
- FW: contains the IP Phone downloadable firmware files
- DATA: contains the tone, cadence, and IP Phone gain tables

- LOCALE: contains the LANGUAGE.INI file for IP Phone country-specific information
- ETC: exists only if created to store a startup script

### Reformatted root directories

If a VGMC has the C: drive erased, then the VGMC application rebuilds the necessary directories. The following is the top level directory structure, with only the directories built by the VGMC application.

```
-> ll size date time name
-------- ------ ------ --------
512 JUN-06-2000 08:28:44 LOG <DIR>
512 JUN-06-2000 08:28:46 OM <DIR>
512 JUN-06-2000 08:28:50 CONFIG <DIR>
512 JUN-06-2000 08:33:34 FW <DIR>
512 JUL-01-2000 12:00:22 DATA <DIR>
512 JUL-01-2000 12:00:22 LOCALE <DIR>
value = 0 = 0x0
->
```

The directories are built as part of the VGMC application boot sequence. However, the /FW directory is built by the UMS task. This task (and many others) does not start until after communication is established with the Call Server. Communication with the Call Server requires the CONFIG.INI file and BOOTP data to be on the card. This means you cannot use OTM to download the IP Phone firmware file (C:/FW/fwfile.1) to the card until you download the CONFIG.INI file and BOOTP data and reboot the card. The firmware download fails if you try before completing the other steps. The NTP specifies the correct order of installation as follows:

1. Download node and card properties and new VGMC application if needed.
2. Reboot card and verify Call Server link is established.
3. Download IP Phone firmware file.

If the PBX link cannot be established or OTM is not available (M1/CSE1.1 only), perform the following process:

1. Manually create the C:/FW directory (`-> mkdir "FW"`)
2. Transfer, using FTP, the previously saved compressed firmware file to the card (/FW) (for example, use `hostFileGet`).
3. Use the command swDownload or swCopy to load the new VGMC software to the card, FTP the file to the card first if necessary.

4. Download the node and card properties from an FTP server (using **`bootPFileGet`**, **`configFileGet`** or **`hostFileGet`**).
5. Reboot the card.

## Signaling Server

Unlike the VGMC /C: root directory, the LTPS on the Signaling Server uses two root directories to store files: /p and /u. The directories typically found on the VGMC C: root are under the /u directory on the Signaling Server. For instance, the configuration files are stored in the /u/CONFIG directory.

The following example shows command output.

1. /p Directory

```
-> 11
size date time name
-------- ------ ------ --------
512 NOV-20-2002 17:31:34 DATA <DIR>
512 NOV-20-2002 17:31:34 ETC <DIR>
512 NOV-20-2002 17:31:34 GK <DIR>
512 NOV-20-2002 17:31:34 INSTALL <DIR>
512 NOV-20-2002 17:31:34 LOAD <DIR>
512 NOV-20-2002 17:31:34 WEB <DIR>
25219 NOV-20-2002 17:31:36 DISK.SYS
271511 NOV-20-2002 17:31:36 MAINOS.SYM
1687448 NOV-20-2002 17:31:38 MAINOS.SYS
512 NOV-20-2002 17:43:48 NVRAM.SYS
0 NOV-20-2002 17:31:38 RENFILE.SYS
0 NOV-20-2002 17:31:38 RESFILE.SYS
527360 NOV-20-2002 17:34:28 BOOTROM.SYS
```

2. /u Directory

```
-> 11
size date time name
-------- ------ ------ --------
512 NOV-13-2002 16:40:34 ODS <DIR>
512 NOV-14-2002 11:11:46 DATA <DIR>
56 NOV-13-2002 16:45:20 DISKTEST.LOG
512 NOV-13-2002 16:58:34 CONFIG <DIR>
512 NOV-13-2002 16:58:34 DB <DIR>
512 NOV-13-2002 16:58:34 FW <DIR>
512 NOV-13-2002 16:58:34 GK <DIR>
512 NOV-13-2002 16:58:34 LOG <DIR>
512 NOV-13-2002 16:58:34 PATCH <DIR>
512 NOV-13-2002 16:58:34 RPT <DIR>
```

```
512 NOV-13-2002 16:58:34 TMP <DIR>
512 NOV-13-2002 16:58:34 TRACE <DIR>
512 NOV-13-2002 16:58:34 WEB <DIR>
512 NOV-14-2002 11:11:46 OM <DIR>
512 NOV-14-2002 11:11:46 LOCALE <DIR>
512 NOV-20-2002 17:31:34 NVRAM.BAK
value = 0 = 0x0
```

3. /u/CONFIG Directory

```
-> 11
size date time name
-------- ------ ------ --------
512 NOV-13-2002 16:58:34 . <DIR>
512 NOV-13-2002 16:58:34 .. <DIR>
598 NOV-21-2002 10:42:58 BOOTP.TAB
2117 NOV-21-2002 10:43:06 CONFIG.INI
598 NOV-21-2002 14:43:18 BOOTP.BAK
611 NOV-25-2002 08:59:34 UMS.INI 10025
NOV-25-2002 08:59:42 CONFIG.VAL 2117
NOV-21-2002 10:43:04 CONFIG.BAK 44
NOV-21-2002 13:00:02 SECURITY.INI 30
NOV-25-2002 11:20:10 USERDATA.INI
```

# Troubleshooting

This section contains tips and suggested commands to troubleshoot problems.

## Network sniffer

A network sniffer is an excellent tool for troubleshooting network devices. By capturing exactly that information that is sent on the wire and providing decoders for the various protocol levels, capture traces often provide solid answers about why phones do not register, why Voice Media Gateway Cards (VGMC) do not communicate, or the source of voice QoS problems.

### Connecting a sniffer

A hub is useful to connect the sniffer at the device being investigated. With a hub, the sniffer sees the same traffic as the VGMC, Signaling Server, or IP Phone because all ports receive the same data. Match the hub speed to the device speed. Although not recommended, you can use a 10BaseT hub on the VGMC or Signaling Server TLAN. However, Nortel does not recommend that you place the TLAN on a 10BaseT hub if many IP Phones register with it or when high concurrent usage exists for gateway ports.

If you use a smart hub, devices connected at 10BaseT cannot detect traffic generated by devices connected at 100BaseT and vice versa. Therefore, you must configure your sniffer to use the same speed as the device you want to sniff to capture packets.

When a Layer 2 switch is used, you must configure port mirroring (or similar product depending on your product) so the switch can send a copy of all packets on a port to the port with the connected sniffer.

If the Layer 2 switch does support port mirroring, or you have no access to the management interface, you can insert a passive hub in line with the desired port and connect the sniffer to it.

It can be useful to connect a sniffer at both ends of an IP Phone to VGMC or Signaling Server connection that has problems. Compare the two capture files to uncover network problems, such as a router discarding signaling messages or old ARP cache addresses in packets.

The MGC and MC32S cards can mirror a port without using an external hub or switch.

## Collecting sniffer captures

To facilitate analyzing and cross-referencing sniffer captures, ensure the PC, on which the sniffer runs, has the same date an time on the device being sniffed. Because hundreds of packets can be generated within a few seconds, synchronizing the time to the second is recommended.

Use the capture filters to reduce the amount of captured information (unless capturing everything can provide clues to the root of the problem). For instance, create a filter with the VGMC or Signaling Server TLAN interface and the IP address of the IP Phone being debugged. First, create a new profile. Then modify the address parameters of the profile for the VGMC/Signaling Server TLAN and the IP Phone address. Finally, select the filter before you start the capture.

> ***Note:*** Be sure to select IP for the Address Type. If set to Hardware, the MAC address of the endpoints is expected but the program allows you to enter an IP address; you receive no indication of any problem until you receive an "Invalid Hardware Address" error when you try to capture using the filter.

Use the display filters to filter information from a capture file while you diagnose a problem. A new window appears each time an applied filter changes the contents. You can save only those windows with useful information. The process create the filter and selecting it is the same process you use for the capture.

After you stop the capture, display it. Select the Decode tab on the expert window to see the packet information. If you installed the UNIStim or RUDP packet decode package, the contents of the UNIStim and RUDP messages are decoded as well as all default protocols (IP, TCP, and RTP).

The Matrix tab shows the communication between various devices as detected by the sniffer. On the Matrix tab, you can filter by selecting an IP address and clicking the Filter button. This procedure shows all traffic send and received by the device that uses the selected IP address.

The Host Table lists statistics for each host in the capture, while the Statistics tab shows overall statistics.

Ping and Traceroute are available on the Tools menu.

When no host names are assigned to network devices, you can create names using the Address Book command on the Tools menu. The names then appear in the capture files in place of the IP address so you can read the trace files faster than reading IP addresses. Save this address book under the Address Book command on the Database menu.

To debug problems related to timing, you can select a packet in the Decode tab Summary Section; right-click and select Mark Current Frame. Notice that the selected packet has a Relative Time of 0 seconds to provide an easy way to determine the amount of time that passed since a particular event. This is helpful to debug IP Phone reset problems.

## VoIP problems

This section describes VoIP problems you may encounter.

### VGMC IP addresses incorrectly configured

If the VGMC detects a duplicate IP address, the card may print one or more of the following messages:

```
JAN 02 08:06:57 tNetTask:  Info arp info overwritten for
c0a80195 by 00:60:38:01:  a1:46
JAN 02 08:06:57 tNetTask:  Info duplicate IP address
c0a80195 sent from Ethernet address 00:60:38:01:a1:b8
```

The same IP address for two devices can cause unexpected and random behaviours on the IP Phones or the VGMC or Signaling Servers. You must identify the device that is configured to use the displayed IP address, locate the device specified by the MAC address, and resolve the duplicate IP assignment.

### VGMC receiving incorrect Bootp information

Problems can occur when VGMCs receive invalid information from a BOOTP or a DHCP server. The result can be a corrupt leader card. The workaround is to turn the BOOTP relay agent off at the router where the ELAN segment is connected.

### IP Phone goes offline

Two scenarios can cause the IP Phone to go offline. Each can be identified by logged messages.

#### Watchdog reset

The TPS periodically sends a watchdog timer reset message to every IP Phone (see “Watchdog reset” (page 29)). This message resets a watchdog timer on the IP Phone and causes the IP Phone to send an acknowledgement. If no response is received from the IP Phone, the RUDP transport resends the message up to 10 times, waiting 400 milliseconds between each for a response. After the tenth retransmission, the TPS prints an error message indicating that the RUDP link failed, marks the telephone as offline, and notifies the Call Server core.

When the IP Phone fails to receive the watchdog timer reset message, the watchdog timer times out, reboots, and begins registration. This occurs when a network fails or high traffic through a router causes the messages to be lost or delayed. This can happen when the IP Phone is on a subnet different from the TPS TLAN and one of the network routers drops packets. The RUDP polling message is a plain UDP message and is thus a candidate for being dropped before higher priority packet data. Another possibility is that the IP Phone remains unplugged while the TPS card polls it.

The following is an example of the messages printed on the Maintenance port of the TPS card for this scenario:

```
JAN 23 15:32:59 tRDP: Error ITS2008 Terminal connection
status:  192.168.1.141 lost (20)
JAN 23 15:32:59 tVTM: Notice 192.168.1.141 Unregistered,
terminal = 0x3a86e64, device = 0x3a86ff8
JAN 23 15:32:59 tSET: Info Terminal offline 192.168.1.141
TN 0x6005
```

Then when the telephone reboots and registers, the following prints:

```
JAN 23 15:36:50 tCSV: Info 192.168.1.141 Connecting to node
1, TN: 61.1, 0x6005
JAN 23 15:36:52 tVTM: Info ITS5008 Terminal connection
status:  192.168.1.141 ok (20)
JAN 23 15:36:53 tSET: Info 192.168.1.141 TN 61-01
Registered with M1
```

### IP Phone reboots or power cycles between pollings

If the IP Phone reboots or has power cycled between pollings, the TPS does not detect the failure but receives an unexpected registration. Because the TPS does not determine that the IP Phone was offline, a different sequence of messages print:

```
JAN 23 15:39:37 tCSV: Info 192.168.1.141 Connecting to node
1, TN: 61.1, 0x6005
JAN 23 15:39:39 tRDP: Warning 192.168.1.141 Connection
restarted, cid = 0x33cdb28
JAN 23 15:39:39 tVTM: Notice 192.168.1.141 Unregistered,
terminal = 0x33cd74c, device = 0x33cd4b4 JAN 23 15:39:39
tSET: Info Terminal offline 192.168.1.141 TN 0x6005
JAN 23 15:39:39 tVTM: Info ITS5008 Terminal connection
```

```
status:  192.168.1.141 ok (20)
JAN 23 15:39:40 tSET: Info 192.168.1.141 TN 61-01
Registered with M1
```

### Sniffer captures of IP Phone resets

The following are tips for collecting sniffer captures of IP Phone communications with the TPS:

- On the CS 1000 and CS 1000M systems, the Signaling Server has priority over the VGMC TPS. For nodes with Signaling Server as the leader, the Signaling Servers require that all IP Phones on the node be registered. When a primary and secondary Signaling Server are on a node, the telephone registrations are split between them. If you are unsure which Signaling Server a telephone registers to, you can disable the LTPS on the other Signaling Servers by using the disiTPS CLI command (if sufficient registration capacity is available on the remaining LTPS for the displaced phones).

  ***Note:*** If the TLAN connection to the available Signaling Servers fails, phones can register to the VGMCs. The phones do not automatically switch back. You must enter the **`loadBalance`** CLI command after the Signaling Server returns online to return the phones to the Signaling Server (see "loadBalance" (page 195)).

- On the Meridian 1, if you have multiple VGMCs in a node with approximately the same number of phones for each card, you cannot be certain which card uses new telephone registrations. However, you can reboot one of the VGMCs so that it has no registered phones. Because the telephones will register with the VGMC that has the least number of phones registered, you know in advance that this card uses new IP Phone registrations.

  If you have a specific IP Phone (or group of phones) that you want to capture information on, you can register the telephones to that card. If IP Phone reset problems occur, you know that the IP Phone registers to the same card (provided the number of phones registered to the card is not equal to the number on other VGMCs). Consequently, you need not connect a sniffer to every card on the node to capture the problem.

- When you work with IP Phone reset problems, try to register the telephone you are monitoring to the Leader card. This way, you can see the entire registration process (Connect Server messages appear).

- Ensure that the times on the Sniffer PCs are synchronized with the time on the TPS. To determine the date and time on the TPS, through the vxWorksShell, run the following command:

```
-> date
```

Now analyzing the sniffer capture is easy. Ensure that the buffers on each sniffer are configured to wrap around and write to file.

- When sniffing a LTPS card, try to capture the CLI output at the same time. Use the same sniffer PC (that uses the COM port and plugs into the faceplate). Retrieve log files from the LTPS cards for cross-referencing.
- You can perform port mirroring, using the ethportmirror command, to connect LAN analyzer equipment to the Layer 2 switch and capture LAN traffic on external LAN ports (such as Layer 2 TLAN/ELAN and 100BaseT Media Gateways). You can also use port mirroring to capture the Signaling or proprietary message traffic (Mindspeed Tone and Conf Module, Expansion Boards, and VoIP Daughter Boards) between internal Layer 2 components. For more information about using the ethportmirror command, see "“ethportmirror” (page 328)".

## Reset conditions for IP Phones

If an IP Phone resets, it stores the reason for the reset in its nonvolatile memory. The information prints from the maintenance port of the connect Server and in the log file when the IP Phone registers again, as follows.

```
31/10/02 17:39:41 LOG0006 CSV: 192.168.20.11 Connecting to
node 7812,

TN: 61-01, 0x6005 [Soft Reset:  Watchdog timeout, code:  1]
31/10/02 17:39:50 LOG0006 CSV: 192.168.20.12 Connecting to
node 7812,
TN: 61-00, 0x6004 [Soft Reset:  Watchdog timeout, code:  1]
```

Use the “usiQueryResetReason ” (page 295) command to retrieve this information. The data is valid only until the next reset occurs, which overwrites the prior value.

## Reset conditions for the IP Softphone 2050

The IP Softphone 2050 resets if one of the following conditions occurs:

- The watchdog timer expires.
- The IP Softphone 2050 is instructed to hard or soft reset by the CS 1000.
- IP Softphone 2050 change.
- Firewall applications that intercept or delay packets, cause the IP Softphone 2050 to reset because the Watchdog timer expires.

Exercise care when you use firewall applications on the same computer as the IP Softphone 2050.

Watchdog timers can expire when there is no UNIStim traffic on the IP Softphone 2050.

### Signaling Server or VGMC Watchdog timer values

Depending on which LTPS platform the IP Phone registers with, the watchdog timer value varies. The following table shows the time values.

| LTPS platform | Watchdog timeout (seconds) | Maximum time between reset messages received by a telephone (seconds) |
|---|---|---|
| Signaling Server | 200 | 100 |
| ITG-P | 192 | 96 |
| SMC | 256 | 128 |

## IP Softphone 2050 not connecting to TPS

If the IP Phone displays the message "Connecting...," "Server unreachable. Reconnecting in ...nn seconds," or "· 802.1 QoS is not supported on your network," the IP Softphone 2050 tries to open a UNIStim connection (socket) with the CS 1000. If the CS 1000 is up, reachable, and the IP Softphone 2050 uses the correct IP address, the message "Connecting..." appears only briefly and the message "Server unreachable..." does not appear. The latter message indicates a problem reaching the CS 1000:

- Check the connectivity of the PC to the network. Try to ping another station on the network that you know is up. Try to ping the CS 1000. Check if other stations are up.
- Verify the IP Softphone 2050 server address (IP and port) in the IP Softphone 2050 configuration utility. Verify CS 1000 configuration to ensure the IP Softphone 2050 request to connect is permitted (enough ports are programmed, security is operational). In the IP Softphone 2050 configuration utility on the QoS tab, select Off, and restart the IP Softphone 2050. Use of the trace utility or a sniffer may yield more information about the problem.

A message "Connection Established. Reinitializing..." indicates that the IP Softphone 2050 established a connection (socket) with the CS 1000. The IP Softphone 2050 has sent a request to the CS 1000 to resume connection. The CS 1000 is expected to reply to the this request. During startup, the Call Server passes the IP Softphone 2050 to various subcomponents, causing this message to rapidly appear two or three times during a normal startup. If the message is not a transient startup message, then review the CS 1000 programming. The number of times the IP Softphone 2050 passes to other subcomponents can help you identify the problem. Use the trace utility or a sniffer to debug the problem.

## IP Phone not connecting to TPS

If an IP Phone continuously registers but then reboots after the watchdog timer period expires, check for an incorrect VLAN configuration. This can occur when the Enable 802.1Q support check box is selected but the IP Phone is on a Layer 2 switch that does not support VLAN tags or is not configured. The IP Phone initially registers to the Signaling Server or VGMC because it starts with no VLAN tagging; the TPS then enables the VLAN tagging because you enabled VLAN priority. Because the switch does not expect the VLAN tagged packets, which then start to arrive from the telephone, it discards them. The LTPS no longer can communicate with the IP Phone so the IP Phone eventually resets when the watchdog timer expires.

To correct this problem, either deselect the Enable 802.1Q support in EM/OTM or configure the network for 802.1Q VLAN tagging.

## VGMC or Signaling Server locks up

If a VGMC or Signaling Server card locks up (for example, you cannot communicate with the VGMC through the faceplate or a Telnet session, you cannot use gateway channels, or telephones cannot register to the card), use the following techniques to diagnose the card:

- Note the faceplate display. If an error code appears, check the applicable NTP and perform the suggested steps.
- Attempt to ping the TLAN of the locked card from another device on the same subnet (for example, another VGMC)
- Attempt to ping the ELAN of the locked card from the Call Server or the OTM/PC.
- Check if the VGMC 8051 processor responds. From the Call Server, enter the following commands:
  - — LD 32
  - — idc <card TN>
- If you receive a response from the command in the previous step, then check if the processor responds. From the Call Server, enter the following commands:
  - — LD 32
  - — disc <card TN>
  - — enic <card TN>

- If you get a response from the command in the previous step, then, if applicable, check if an IP Phone currently registered to the VGMC/Signaling Server is responding:
  - — LD 32
  - — idu <IP Phone TN>
- If al the preceding steps provide a response, then your card is not locked. Perhaps your shell task is locked. Connect to the serial port (faceplate or octopus cable), if necessary, and press Ctrl+Q to see if you can access the card. Certain tasks may be suspended.
- If all the preceeding steps fail to bring up the card, then you must reboot the card.
- If a reboot fails, then unseat and reseat the card. After you reboot the card, retrieve the log files from the VGMC/Signaling Server, and the history file and report log from the Call Server, and send them to your support personnel for analysis.

## VGMC continuously reboots and downloads firmware versions

If you have multiple cards in your node and an IP Phone constantly reboots and downloads firmware, then check that each card has the same firmware version. Use the command "umsPolicyShow" (page 284) to see which firmware will be downloaded to a telephone. Only one firmware version can be installed on a node; after a telephone downloads new firmware, it reboots and registers again. If it registers to a TPS with a different firmware version, it downloads that new firmware and then reboots. This process can continue indefinitely if all cards on the VGMC node have approximately the same number of registered telephones.

## Incorrect IP Phone software version

If IP Phones are not downloaded with the expected version of firmware, the firmware file may be missing from the card or have an incorrect file name. The expected file name is i2002.fw for the IP Phone 2002 and i2004.fw for the IP Phone 2004. Use the command "umsPolicyShow" (page 284) to see which firmware will be downloaded to a telephone. Check the /FW directory to ensure that a firmware file with the valid file name is present.

```
IPL> umsPolicyShow

Total firmware = 1

FirmWare Retry TermType PolicyName Server FileName Limit
When Upgrade Protocol
--------- --------- ------------ -------------- ---
------------- ----- --------- --------- --------------
```

```
-----
0602B59 -1 i2002 DEFAULT_I2002 192.168.1.140 /ums/i2002.fw
10 ALWAYS ANY TFTP

value = 0 = 0x0
IPL>
-> 11
size date time name
-------- ------ ------ --------
512 JUN-06-2000 08:33:34 .  <DIR>
512 JUN-06-2000 08:33:34 ..  <DIR>
933763 JUN-06-2003 08:33:38 i2002.fw
950810 JUN-06-2003 08:37:22 FWFILE~1.1

value = 0 = 0x0
```

## VGMC link to Call Server fails

If the link to the Call Server fails, the VGMC application prints the following message. An SNMP alarm (ITS1009) occurs and displays on the VGMC faceplate.

If the IP address of the Call Server changes, you must initialize the Call Server to apply the changes. Also make sure that the ELAN Configuration section in the config.ini file of the VGMC is correct.

```
->
192.168.1.10 (192.168.1.10) deleted

JAN 23 15:54:33 tRDP: Alert ITS1009 Call server
communication link:  192.168.1.10 down (20)
JAN 23 15:54:33 tpbxReset:  Info Attempting to establish
PBX link
JAN 23 15:54:33 tTPS: Info ELAN connection down, refuse
further registration
JAN 23 15:54:33 tVGW: Info GW OffLine msg received from
pbxLib, close all dsp channels and unregister Gateways
after 600 seconds
JAN 23 15:54:33 tSET: Info PBX Link down, reset all
registered set after 600 seconds
```

When the link recovers, the following prints.

```
JAN 23 15:55:48 tpbxReset:  Info ITS5009 Call server
communication link:  192.168.1.10 up (20)
JAN 23 15:55:48 tpbxReset:  Info PBX UDP link established
JAN 23 15:55:48 tpbxReset:  Info PBX TCP link not used
JAN 23 15:55:48 tTPS: Info ELAN connection up, accept set
```

```
registration
JAN 23 15:55:48 tVGW: Info GW OnLine msg received from
pbxLib
JAN 23 15:55:48 tVGW: Info vgwSyncReqReceive:  freq 0,
rlsCall 1, callServer 0
JAN 23 15:55:48 tSET: Info dsetSyncReqReceive:  freq 0,
rlsCall 1, callServer 0
JAN 23 15:55:48 tCSV: Info Callserver type set to Meridian
Small system
JAN 23 15:55:48 tVGW: Info Channel 0, already registered
with M1 (?)
JAN 23 15:55:48 tVGW: Info Channel 1, already registered
with M1 (?)
...  (one msg per channel)
JAN 23 15:55:48 tVGW: Info Channel 23, already registered
with M1 (?)
JAN 23 15:55:48 tSET: Info 192.168.1.142 TN 61-02
Registered with M1
...  (one msg per phone)
JAN 23 15:55:48 tSET: Info 192.168.1.141 TN 61-01
Registered with M1
```

## Call connects through VoIP gateway with no speechpath

If a call connects through the gateway but no speechpath exists, check the DSP receives data from the DS-30X backplane.

- If the problem call involves a trunk or far end VGMC or IP Phone, isolate the call down to one system, by conferencing in a local digital telephone. If the one-way speechpath still exists, then you can eliminate the far-end telephone or VGMC as being the source of the failure. Alternatively, if you can duplicate the problem, initiate a call between the IP Phone and a TDM telephone.
- Enter the command "tsm_stat_req_tele_levels " (page 275) for the channel; the levels are typically in the –300 to –600 range.
- Press and hold a digit key on the TDM telephone to generate DTMF towards the VGMC.
- Enter the command again and, if the levels are in the –10 to –100 range, then PCM passes between the card and the system. Otherwise, a problem likely exists with the card.
- Use the command "genToneOn chNum, side, freq, duration" (page 131) to generate a tone from the DSP to the TDM or from the DSP to the IP Phone. This can isolate an invalid network connection from the IP Phone to the DSP or a bad timeslot or timeswitch connection from the TDM network to the DSP.

#### Calls to IP Phones in other VGMC nodes with separate TLANs

If multiple VGMC nodes belong to separate TLANs, but are configured to the same customer and use the same zones, then the potential exists for a one-way speechpath during IP to TDM calls or IP to IP calls across nodes. This situation occurs because the Call Server has no concept of a VGMC node; all IPTNs that belong to the same Call Server Customer Number are treated as one resource pool, regardless of the node ID. Therefore, the Call Server can choose any channel that belongs to the customer.

The selected channel can belong to a card in a different node. If the TLANs are separate and the IP network cannot route between the two subnets, then the IP Phone cannot access the IP address associated with the channel, which results in no speechpath.

If the IP network can route calls between the two subnets, then make sure that you correctly configure the default gateway. If no routing capabilities exist between the two subnets, you can solve the problem by adding a new zone for each node on a separate TLAN, and then configuring the channels and IP Phones to belong to the new zone.

### Cannot make calls with particular zone and codec configurations

A problem can exist where a call connects but the codec does not match your expectation based on the zone configuration. Or, a call can fail to connect, and an error message prints on the Call Server TTY stating "No codec match found". This error occurs if you do not download the card configuration to all cards in the node after modification of the codec list. Then, when an IP Phone registered on one VGMC/Signaling Server calls an IP Phone registered on a different VGMC/Signaling Server or makes a gateway call by using a channel from a different VGMC, a codec mismatch can occur.

The codecs configured in EM/OTM are placed in the CONFIG.INI file (see “CONFIG.INI” (page 408)) and downloaded to the VGMC. When the VGMC gateway channels register with the Call Server, the configured codec capabilities are sent to the Call Server CPU. Likewise, when an IP Phone registers with the system, the list codecs reported by the telephone as supported are sent to the Call Server. When a call occurs, the Call Server decides which codec the call uses based on the zones of the two endpoints:

- If the call is within the same zone (that is, both have the same zone value configured), the Intrazone codec is used; otherwise, the Interzone codec is used.
- If a mismatch occurs between the codecs (that is, one endpoint has BB (for example, G.729AB) and the other BQ (for example, G.711)), the

Call Server chooses the codec with the lowest bandwidth usage (that is, BB).

- In the case of a gateway call, the zone of the VGMC gateway channel is compared to the IP Phone zone.

A problem can occur when the codecs change and the change is not downloaded to all cards in the node. This can result in the two endpoints registering with different codec lists. If only one codec matches between the two lists, the call may complete but use a codec different from the codec expected. When no codecs match between the two lists, the call fails and an error message prints on the Call Server stating that no codec match can be found.

Check this condition by entering the command "rlmShow " (page 375) at the Call Server pdt prompt. This prints the codecs configured on all telephones and gateway channels. If a configuration problem occurs, different lists for different phones and gateway channels print; ensure that the list is identical for every device.

In summary, to avoid errors whenever card properties in the OTM change, you must download the change to all cards in the node. If this is done, this problem won't occur.

## No dial tone on IP Phones when registered to Call Server with dual CPU

If your phones are registered but you receive no dial tone, check that the VGMC to which your telephone is registered is talking to the active Call Server CPU. On the VMGC, use the **`rudpShow`** and **`pbxLinkShow`** commands (see "rudpShow" (page 232) and "pbxLinkShow" (page 220)) to verify that the card communicates with the proper CPU. On the Call Server, you can check the network configuration by using the LD 117 **`prt elnk`** command.

## Input errors on the switch side or connection failure

If you find that you receive numerous input errors on the switch side or if a connection fails, it may be due to an ELAN/TLAN port misconfiguration between the expected speed and duplex and the actual values.

The Media default settings are as follows:

ELAN: 10M/Half
TLAN: Auto-negotiate

Therefore, if the switch ports are Auto-negotiate, which is the recommended setting, then the ports should show

ELAN: 10M/Half
TLAN: 100M/Full

However, some cards appear where the ports are configured as follows:

ELAN: 100M/Full
TLAN: 10M/Half

To correct the problem, perform the following steps:

1. Issue the following command from the vxWorks shell:
   nvramTLanSpeedSet 10100
   nvramELanSpeedSet 10
   nvramELanDuplexSet 1
2. Reboot the card.

Several methods are available to check the connections speed and duplex:

- faceplate LEDs
- itgCardShow
- from vxWorks, linkGetOperation 1 or 0

```
> linkGetOperation 0
value = 100 = 0x64 = 'd'

NOV 26 08:42:17 tShell:  Info T-LAN in Autonegotiation
Mode.

NOV 26 08:42:17 tShell:  Info T-LAN Operating in 100Mbps,
Full-Duplex mode.

-> linkGetOperation 1
value = 10 = 0xa

->

NOV 26 08:42:40 tShell:  Info E-LAN Operating in 10Mbps,
Half-Duplex mode.
```

## New VGMC software or IP Phone firmware testing

To try a new VGMC version or a new IP Phone firmware version on just a few telephones, or to perform testing at the customer site without affecting all users, you can create a new node. You must add another card (or reconfigure an existing card) to the system but assign a new node number that is different from the existing node number. Configure the card as a Leader because each node must have a leader. You can then reconfigure selected IP Phones to register with the new card (that is, change the

S1/S2 address and node number on the IP Phones). When the IP Phone reboots, it registers with the new node. All the unmodified IP Phones continue to use the existing node.

When you complete the testing, remove (or reconfigure) the card, and reconfigure the IP Phones to the original settings.

Configure the gateway channels and IP Phones associated with the new node to use separate zones from the original node, if the two nodes are on separate TLANs with no routing capability in between (see "Calls to IP Phones in other VGMC nodes with separate TLANs" (page 38)).

## IP Phone commands

This section describes IP Phone commands.

### Maintenance telephone

A digital telephone functions as a maintenance telephone when you define the class of service as MTA (maintenance telephone allowed) in the Multi-line Telephone Administration program (LD 11). You can use a maintenance telephone to send commands to the system, but you can use only a subset of the commands you can enter from a system terminal.

To access the system by using the maintenance telephone, enter a SPRE code (defined in the customer data block) followed by 91. Then enter the overlay commands. To enter commands, press the keys that correspond to the letters and numbers of the command (for example, to enter LD 42 Return, press 53#42##). The following overlays are accessible from an IP Phone that operates as a maintenance telephone: LDs 30, 32,33, 34(except for TDS commands), 35, 36, 37, 38, 41, 42, 43, 45, 46 (except TONE commands), 60, 61, and 62.

### VGMC node TN password functionality

Basic craftsperson node-level TN entry password protection exists on the IP Phones to control registration with a virtual line TN on the Call Server.

When the password is configured and enabled, instead of the IP Phone displaying node ID and TN fields, the screen shows the four-digit node ID and a password prompt. After you enter the node ID and password and press OK, if the password passes the Connect Server authentication, a screen appears with the TN field. If you do not enter the node ID and password are not entered, the registration continues after 5 seconds and the TN never appears.

If you enter an invalid node ID password is entered, the node ID and password screen reappears. This screen reappears a maximum of two times, to provide three opportunities to enter the password. After three attempts, registration continues as if no entry occurred at the IP Phone. You can reboot the IP Phone and try again, if necessary.

If you enter a zero length password, then the node ID, TN and password screens do not appear on the IP Phone during registration. This provides maximum security to prevent any entry of a password or TN from the IP Phone.

In addition, you can enter a temporary password. Two parameters for the temporary password determine whether it expires based on the number of uses or after a period of time. The temporary password is automatically deleted after it is used the defined number of times or when the duration expires, whichever occurs first.

When the craftsperson node-level TN entry password protection is enabled, the Set Info submenu of the Telephone Option menu no longer displays the Set TN, node IP, node ID, and IP address of the TPS to which the telephone is registered. You can retreive the TN of the telephone on the CS 1000 through the LD 20 **PRT DNB** command and LD 32 **IDU**, or LD 80 **TRAC**, or **PDT> rlmShow**. You can also find the TN on the ITG Line card by using the **ITGL> isetShowByIP** command.

When you install a VGMC node, you define no password or temporary password and the password feature is in the disabled state. If you enable the password before you configure the node password, the password protection is enabled with a null password (so the password and TN prompts never appear on the IP Phones).

The password and password protection status is stored in the SECURITY.INI file in the VGMC /C:/CONFIG or the Signaling Server /u/CONFIG directory. This file is created when you enter the **nodePwdSet** or **nodePwdEnable** commands the first time. The file remains in the directory until you reformat the C: drive or delete the file. To provide redundancy, the SECURITY.INI file exists on every card in the node. Whenever the password or status changes, it is sent to all cards in the node. Each card then updates its SECURITY.INI file with the new information. If the file does not exist, it is created. When a card that is not the Master boots, it retrieves the password and status from the Master. If the SECURITY.INI files does not exist, the card creates it with the information retrieved from the Master. Only the first card that becomes the master by default retrieves the password and status from the SECURITY.INI file.

The temporary password is not saved to the file. It is retained in memory on all cards in the node. If a card reboots, it retrieves the temporary password from the Master. If all cards on the node do not reboot at the same time, the temporary password is retained. However, if all of the cards reboot at the same time, the temporary password is lost; you can then enter a new temporary password, if required.

## IP network troubleshooting

This section describes IP network troubleshooting.

### General troubleshooting suggestions

Because the IP Phone depends on the IP network to communicate with other IP Phones, problems on the LAN or WAN can cause a variety of voice quality and usage issues. The following are some suggestions to determine if the network is causing problems:

- Make sure the ELAN, TLAN, and node IP addresses are properly configured on the VGMC or Signaling Server. Every card has unique ELAN and TLAN interface IP addresses. The node IP is shared among the cards and is programmed on the TLAN interface of the current Master.

  ***Note:*** Use separate subnets for the ELAN and TLAN interfaces.

  — ELAN (Embedded LAN): carries maintenance, administration, and alarm data between OTM/EM and the VGMC/Signaling Servers. It also carries RUDP/TCP signaling traffic between the Call Server and the VGMC/Signaling Servers. All ELAN addresses for all nodes must be on the same subnet and must be the same as the Call Server CPU subnet.

  — TLAN (Telephony LAN): carries RUDP and RTP packet data between VGMC/Signaling Server and IP Phones. The TLAN addresses in all devices in a node must be on the same subnet.

  — CLAN (Customer LAN): regular customer LAN for PCs. Protect the VGMC/Signaling Server TLAN from the broadcast traffic that can regularly occur on the CLAN by isolating it to its own VLAN. Turn off Spanning Tree, or configure ports as fast port enable, fast learning.

  — Node IP (on the TLAN subnet): used by IP Phones to register with the node. This address is shared by the devices; it is assigned to the TLAN interface of the current Master.

- Check the configuration of the switch or router ports connected to the VGMC/Signaling Server and IP Phones (see “TLAN packet loss errors” (page 45)).

- Check the faceplate indicators on the VGMC or CPPM Signaling Server cards as follows:
  - ITG-P:
    - Is the green carrier detect LED on (indicating TLAN is connected to hub)? Some early ITG octopus cables reversed the ELAN and TLAN cable labels. Unplug both and plug in the TLAN cable; if the green LED does not light, try the cable marked ELAN. Correct the cable labels if they are reversed.
    - Does the yellow activity LED flash when TLAN Receive (RX) traffic occurs to the card?
  - SMC:
    - Is either the 100 or 10 LED on for both the ELAN and TLAN (indicating the respective interface is connected to the switch)? Is the speed as you expect given the configuration?
  - Does the ELAN and TLAN A LED flash when Receive (RX) or Transmit (TX) traffic occurs on the respective interface?
  - Signaling Server:
    - Is the 100 Mbps LED correct for both the ELAN (port 1) and TLAN (port 2)? It is on when the link operates at 100BaseT and off when it is 10BaseT. Is the speed what you expect given the configuration?
  - Does the ELAN and TLAN Link LED flash when traffic occurs on the respective interface?
- Obtain a network diagram of the VGMC/Signaling Server, IP Phones, and all other data devices on the network. Ask the customer how much traffic is on their network. Do router statistics show dropped packets? Try to understand which devices experience problems: all devices or a specific subnet? Do problems occur all the time or intermittently?
- Replacing a VGMC/Signaling Server means a new MAC address (from the new device) is associated with the IP address of the existing device. For some routers, you must ping the router interface from the VGMC/Signaling Server to speed the update of the router ARP table, during which time packets to the VGMC/Signaling Server IP address do not arrive on the new card.
- Try connecting one or more phones and a VGMC/Signaling Server on an isolated LAN to see if the problems persist. For example, using a 10/100BaseT Ethernet crossover cable, connect a single IP Phone to the VGMC/Signaling Server card TLAN interface. If the single telephone works correctly, then a problem likely exists with the LAN. If the problems persist, a card hardware or software problem is likely.

- Run a ping test between the VGMC/Signaling Server or a PC on the same TLAN subnet and one of the phones experiencing a problem. For instance, ping from the VGMC/Signaling Server subnet to an IP Phone once each second for 1 hour. Log the output. Were any packets lost? Was the delay highly variable? These situations indicate that the network may not be suitable for carrying voice traffic.
- Connect a sniffer to the LAN, close to the VGMC/Signaling Server (preferably on a mirrored switch port or a hub) to capture packets going to or from the card. When a problem occurs, stop the sniffer. If the traces and other information, such as the IP address of the devices involved, are provided to support personnel, they can be examined to determine if a network problem exists or if a problem exists with the VGMC/Signaling Server.

  ***Note:*** Potential privacy concerns can occur with this approach because the RTP packets can be reassembled into audio files (a necessary step when investigating voice QoS problems); you may want to notify users about packet monitoring.

- If, after you run the tests, there appears to be a LAN (rather than VGMC product) problem, ask the customer if they can enable any QoS mechanisms in their network. If the problem appears to be with a VGMC product, escalate the issue following the normal escalation process.

## TLAN packet loss errors

The VGMC/Signaling Server software contains detection mechanisms for RTP packet loss. The impact of packet loss varies, but even single lost packets can cause audible clicks, while high packet loss sounds like choppy speech or periods of silence. Packet loss is usually due to a router discarding the RTP packets when the network is busy.

Two types of messages are printed. The first type prints whenever the ITG-P card detects missing packets in the incoming RTP packet stream. The following example warns that two incoming packets were lost at one time for the call on channel 0. This type of detection is not available with the SMC microengine architecture.

```
SEP 22 11:05:49 tRTP: Info SML GAP in RTP seqNo chan 0 (recd: 22802,
expect: 22800, sending 1001, gap 2, tick 98400)
```

The second message type prints a summary at the end of a call showing the percentage of packets lost for that call in each direction (indicated in the message as Receive [RX] or Transmit [TX] from the card perspective). This message also generates an SNMP alarm.

In the following examples, the VGMC determined that 6.4 percent of the incoming (received) packets on channel 0 were lost, while the IP Phone reported 6.6 percent of the outgoing packets from the VGMC were lost. The percentage is of the total RTP packets transferred for the call, so depending on the call duration, the percentage of packet loss can have varying degrees of impact. One to two percent is likely to be noticed and usually 5 percent or more indicates choppy speech at some point in the call.

Example: TLAN packet loss is reported for both IP to IP and IP to TDM calls.

```
SEP 22 11:17:48 tRTP: Warning ITG4028 Voice packet loss:  0
6.4% rx 47.147.75.80 (26)
SEP 22 11:17:50 tVTM: Warning ITG4028 Voice packet loss:  0
6.6% tx 47.147.75.81 (26)
```

In this example, the first parameter (0) is the channel number, if the reported packet loss is in Receive (RX) direction. This parameter is always 0, if the reported packet loss is in the Transmit (TX) direction. The second parameter (6.4 percent) indicates the percentage of total RTP packets transferred that were lost. The third parameter (Receive [RX] or Transmit [TX]) is Receive (RX), if VGMC reports packet loss, or tx, if the IP Phone reports packet loss. If reported packet loss is in the Receive (RX) direction, the fourth parameter (47.147.75.80) is the IP address of the VGMC whose gateway channel is used. If reported packet loss is in the ‘tx‘ direction, the fourth parameter is the IP address of the IP Phone that detected packet loss in the incoming stream.

Perform the following tasks:

- Verify the ports of the router/switch the TLAN and IP Phone are connected to are configured as one of the following (in highest to least desirable order):
    - — Autonegotiation: this is the recommended setting. The VGMC/Signaling Server TLAN interface and the IP Phone autosense the speed and autonegotiate the half or full duplex

setting. Setting the switch or router port to do the same means the fastest possible connection negotiate without intervention.

— Manual setting: 100BaseT, half duplex

— Manual setting: 10BaseT, half duplex

***Note:*** The ITG-P, SMC, and Signaling Server TLAN interface supports 10BaseT and 100BaseT half and full duplex, while the ELAN supports only 10BaseT half duplex. The IP Phones support 10BaseT or 100BaseT half duplex and 10BaseT full duplex.You cannot manually configure a port on the switch or router to 10BaseT or 100BaseT full duplex and have an error-free connection to the IP Phones or VGMC/Signaling Server.

Manual configuration turns off autonegotiation in nearly every product on the market. By definition of the standards, without autonegotiation, the VGMC, Signaling Server and IP Phones revert to half duplex operation. This means the switch or router will be in a full duplex mode while the VGMC device is in a half duplex mode, a situation guaranteed to cause packet loss. If the switch port is set to half duplex and sees lots of late collisions and duplicate collisions or if it is in full duplex mode and sees lots of CRC Errors or runt frames then odds are there is a mismatch.

- Check the router statistics to see the amount of network traffic on each subnet and the number of discarded packets.
- Check for differences in subnet configurations (that is, full versus half duplex or 10BaseT vs 100BaseT) if the problem occurs only for some IP Phones and not others.
- Check which QoS mechanism is enabled on the router to give priority to the packet traffic from the IP Phones.
- To eliminate packet loss messages, configure the switch or router for 10BaseT half duplex operation and check if that eliminates the packet loss messages. If the messages stop, check the wiring and ensure the site has CAT-5 cable.

## Port numbers used by VGMC or Signaling Server application and IP Phones

This section describes the port numbers used by the VGMC or Signaling Server application with IP Phones.

### UDP ports

The VGMC uses the following UDP ports on the TLAN and ELAN interfaces.

**Table 1**
**VGMC/Signaling Server UDP Ports**

| **Interface** | **Platform** | | **Port use** | **Port number** |
|---|---|---|---|---|
| | VGMC | Signaling Server | | |
| TLAN | x | x | TFTP | 69 (used for firmware download to IP Phone 2004) |
| TLAN | x | x | Sun RPC | 111 |
| TLAN | x | x | Syslog | 514 |
| TLAN | x | x | Connection Server signaling | 4100 |
| TLAN | x | x | VTM signaling | 5100 |
| TLAN | x | – | RTP | 5200 + chNum*2: ITG-P: 5200-5246, SMC: 5200-5262 (5200 is the default base voice port number, is configurable through OTM/EM) |
| TLAN | x | – | RTCP | 5200 + chNum*2 +1: ITG-P: 5200-5247, SMC: 5201-5263 (indirectly programmable; based on RTP port) |
| TLAN | x | x | TPS (Node Manager) signaling (with telephones) | 7300 |
| TLAN | x | x | TPS signaling (with other cards) | 16543 |
| TLAN | x | x | SNTP Server | 20000+node id (for example, node 001 has SNTP server port 20001) |
| ELAN | x | x | BOOTP Server | 67 (on Leader card) |
| ELAN | x | x | SNMP | 161 |
| ELAN | x | x | Call Server RUDP signaling | 15000 |
| ELAN | x | x | Call Server RUDP Bcast signaling | 15001 |

The following UDP ports are used by the IP Phone on the TLAN interface.

| Interface | Port use | Port number |
|---|---|---|
| TLAN | signaling | 5000 |
| TLAN | Voice | 5200 (configurable - same as base voice port number on VGMCs) |

***Note:*** For the voice ports, Cisco routers compress RTP headers only when UDP ports are numbered 16384 and higher. Other vendors also look at UDP port ranges to compress. If a customer requires compressed RTP header information, they may have to change the voice port value in OTM/EM.

### TCP ports

The VGMC uses the following TCP ports on the TLAN and ELAN interfaces.

**Table 2**
**VGMC/Signaling Server TCP Ports**

| Interface | Platform | | Port use | Port number |
|---|---|---|---|---|
| | VGMC | Signaling Server | | |
| TLAN | x | x | Sun RPS | 111 |
| TLAN | x | x | FTP | 21 |
| ELAN | x | x | Telnet | 23 |
| ELAN | x | x | Call Server TCP signaling | 15000 |

### SNMP traps

You can use the `itgAlarmTest` or `itsAlarmTest` commands (see "itgAlarmTest " (page 184) and "itsAlarmTest " (page 191)) to verify that the SNMP traps are configured correctly and that the receiving device (OTM or some other trap alarm manager) is receiving the alarms. This test sends the alarm with none of the issues of a real alarm (for example, unplugging the TLAN to generate the TLAN loss of carrier alarm also unregisters the gateway channels and affects call processing). Remember to configure all VGMC/Signaling Servers in the OTM Alarm Notification application so the SNMP traps appear when they occur.

## DSP and voice quality troubleshooting

### DIM error codes

The DIM can print an error message that indicates a problem reported by the DSP. These errors can be channel-dependent (for example, a command to a channel failed), or not (for example, a DSP reset that affects multiple channels). The following two messages can appear depending on the VGMC application version running.

```
JAN 02 11:07:47 tMVX_XSPY: Info
0000753618 - DIM: 0:0 -
MGB_DM_ERROR_INDICATION
(Channel-Dependent):  Error Code:  2 TCID 0


0035475106 - DIM: 2:0 -
MGB_DM_ERROR_INDICATION (Channel-Dependent):  Error Code:
3 TCID 6
```

The message first shows the DSP and channel in the format dsp:channel. It then indicates whether the error is channel-dependent, and then shows an Error Code and, if channel-dependent, the TCID. The error codes are defined as follows:

| Error | Description |
|---|---|
| 0 | MGB_ERR_INVLD_HW_CONFIG |
| 1 | MGB_ERR_INVD_SIG_MSG (invalid message received) |
| 2 | MGB_ERR_INVD_MSG_CHST_CLOSED (invalid message for CLOSED state) |
| 3 | MGB_ERR_INVD_MSG_CHST_IDLE (invalid message for IDLE state) |
| 4 | MGB_ERR_INVD_MSG_CHST_VOICE (invalid message for VOICE state) |
| 5 | MGB_ERR_INVD_MSG_CHST_DTMF (invalid message for DTMF state) |
| 6 | MGB_ERR_INVD_MSG_CHST_FAX (invalid msgmessage for FAX state) |
| 7 | MGB_ERR_INVD_MSG_CHST_TRANSPARENT (invalid message for TRANSPARENT state) |
| 8 | MGB_ERR_TONE_ON_IGNORE |
| 9 | MGB_ERR_INVLD_TONE_PARAMS |
| 10 | MGB_ERR_INVD_CH (invalid channel specified) |
| 11 | MGB_ERR_INVD_ECPATH_COEFF_PARAMS (invalid echo canceller parameters received) |
| 12 | MGB_ERR_PROCESS_OVERLOAD |

## General troubleshooting suggestions

Audio problems can be difficult to troubleshoot because often they are affected by user perception. A card that reboots in a certain situation is perceived as a rebooting card; however, a call with slight distortion or echo may be tolerated by some users while considered intolerable by others. The following suggestions can help you with troubleshooting.

### Echo

Echo is the most frequently reported audio complaint on the VGMC product. The first task is to isolate the scenario in which echo is the experienced.

Ask the following questions:

- What units are involved in the call (for example, IP Phone, TDM telephone, trunk, or PSTN)?
- Does the echo occur on the handset, the headset, or the speaker?
- If the call is transferred to a digital telephone, does the echo still appear? How frequently among the calls made does the echo happen? At what point in the call does the echo happen? What type of trunks does the site have (assuming echo happens on trunk calls)? If multiple types of trunks exist, does the echo occur on all types or only some?
- Does the echo happen for all users or only some? Do commonalities exist between those users (for example, all speak loudly, or all on same subnet)?
- What happens to the echo when the receiver (far end non-IP Phone) of the call unplugs the handset?
- What kind of telephone is involved at the far end?
- Does the echo occur only on calls originated by the IP Phone user or calls received by the IP Phone user?
- When the echo occurs, what is the end caller's phone type (for example, an analog or a digital telephone)?
- Is the echo occurring on local, internal, long distance, or all calls types?
- What is the effect of turning off the sidetone on the IP Phone?

Echo is typically not reported on IP Phone to IP Phone calls. Experience indicates echo usually occurs for calls though the gateway. One indication that the DSP echo canceller is not working optimally is if the divergence count is high (printed at the end of the call when DimECStat is configured or by the tsm_stat_req_ecdbg command). This typically is 15 or less on calls where the echo canceller locks on the echo.

If the echo occurs only as a short burst at the beginning of the call, this is caused by the DSP echo canceller converging on the echo, a normal function. This process typically takes a couple of seconds. It is usually not annoying enough to report, but may be reported with a more severe symptom.

Second, if the echo is occurs only on trunk calls, determine with a non-IP Phone (for example, a 3904 digital telephone) if the same calls have echo. The echo may be caused by trunking, CO, or long-distance carrier problems. If the problem occurs mainly with long distance calls, make a call using a calling card for a different carrier and see if the problem occurs. Analog trunks can cause a complex echo that the DSP echo canceller has difficulty converging on.

**Collecting statistics for echo problems** To collect statistics for echo problems, capture the output of the following commands to a log file (for example, through logged Telnet or HyperTerminal session):

1. Determine the gateway channel being used to make the call that experiences echo:

   — Obtain the IP address and TN of the IP Phone that experiences echo problems (from the IP Phone, use the SERVICES key to retrieve the phone information).

   — Use the TRAK/TRAC commands in LD 80 to determine the physical TN (gateway channel). Alternatively, enter the **`vgwShow`** command (see "vgwShow " (page 309)) on any VGMC in the node and it returns the card and channel the telephone uses.

   After you determine the gateway channel, log on to the associated VGMC and run the **`itgCardShow`** or **`vgwShow`** command. The number printed in the Chan column is the channel number.

2. While the call is in progress, turn the Echo Canceller and DSP statistics on. You must enable the DimDspStat and DimECStat variables to print the output data for the commands in step 3. Exercise caution, because these variables print a block of data at the end of every call for every channel on the VGMC.

   — DimDspStat =

   — DimECStat =

   — dimPrintChannelInfo

3. Run the next set of commands several times during a call experiencing echo. For example, run these commands a few times if the call experiences silence. Then run the commands a few times while the parties talk. Make sure when you log this information, that you document which output corresponds with which scenario. Enter text such as **banner <scenario with which the following output is associated>**. Because no banner command exists, the data is ignored.

   — tsm_stat_req_ecdbg

   — tsm_stat_req_error

   — tsm_stat_req_rx_tx

   — tsm_stat_req_tele_levels

   — tsm_stat_req_vp_delay

   — dimPrintECC

4. Determine the effects of turning the Echo Canceller and Non Linear Processor On and Off. Use the **`tsm_echo_canceller`** command with the appropriate parameters for the following cases:

- ECAN: Off, NLP: On
- ECAN: Off, NLP: Off
- ECAN: On, NLP: Off
- ECAN: On, NLP: On

5. When you finish testing, turn off printing of the ECAN and DSP statistics. Otherwise, the syslog is quickly filled with DSP statistics each time a gateway call occurs.
   - DimDspStat =
   - DimECStat =
6. Send collected statistics to Field Support for analysis.

**Signal limiter** Echo represents the major impairment in VoIP. Although CS 1000 has a carrier-grade echo canceller (ECAN) that complies with the G.168 recommendation from the ITU-T, cases still occur where the existing PSTN network involves nonlinear hybrids. In these cases, the ECAN experiences difficulties eliminating echoes generated when the voice signal is at loud levels.

To solve the problem, Communication Server 1000 uses a signal limiter. The signal limiter (SL) deals intelligently with the voice signal going from CS 1000 system to the PSTN. While preserving the quiet voice signal levels, the signal limiter (SL) adds some attenuation to loud signal levels. Extra attenuation is added when the voice signals become louder.

For versatility, the signal limiter is granular. Several modes of operation exist with only one parameter called SLim. The SLim parameter uses any integer value from 1 to 5. A value of 1 is the most aggressive, while a value of 5 is the least aggressive. A value of 0 means the functionality is disabled, which is the default value when the CS 1000 is installed.

Configure the SLim for each card in the VxWorksShell as follows:

- **`setSLim value`**
  for setting SLim for all channels of the card
- **`tsm_set_slim tcid,value`**
  for specific TCID only when the channel is up

In general, the first command is used.

The more aggressive the SLim is, the more nonlinear distortion is in the transmitted signal.

After you receive a complaint about echoing, perform the following steps:

1. Collect the statistics (see "Collecting statistics for echo problems" (page 52) ).
2. Eliminate all other sources of echo due to improper configuration. Enable the signal limiter.
3. Determine the loss plan and the codec used.
4. If possible, obtain the average voice level transmitted in T1 or E1 links in the PSTN network or a local network if echo is perceived in local analog telephones.
5. Chose an appropriate SLim value for the signal limiter. To start, assign a value of 5 to SLim (least aggressive) and gradually increase the aggressiveness by assigning the SLim value a smaller number until echo disappears.
6. If echo disappears at an acceptable level with a moderate SLim value that does not significantly degrade voice quality, leave SLim at this value.
   If echo disappears at a SLim value where the voice quality is unacceptably degraded, find a compromise between the two sides and set the SLim value accordingly. Adjust the SLim values and watch for echo and voice quality as perceived by the users of the system in hand.

**PCM audio capture** If the call uses G.711, use the Data Capture tool on the VGMC to collect a capture of the audio during the problem.

### Choppy speech
Choppy speech is usually a side effect of network problems, such as packet loss. Packets are lost or arrive late and the DSP must fill in or drop packets. See "TLAN packet loss errors" (page 45)for more information about debugging this type of problem.

### Wavering voice or tones
This problem is typically reported on handsfree calls. Sometimes users report that the receive volume fluctuates during tones. Wavering voice or tones can be caused by an interaction of the telephone speaker and microphone and the handsfree algorithm. To determine the cause, mute the handsfree and press the digits again; the DTMF tones should sound without waver. A similar problem can occur during handsfree conversations when room noise, drafts, or other disturbances cause the handsfree receive volume to fluctuate. Turn down the speaker receive volume to help or eliminate the problem.

## IP Softphone 2050 audio quality

Use the following suggestions to troubleshoot audio quality on the IP Softphone 2050.

### Verifying basic audio operation

To verify the physical connection and proper Windows audio configuration, record an audio message and play it back using the USB headset.

### One-way audio

- Check that the handsfree option is programmed for the IP Softphone 2050 on the CS 1000.
- Check if the mute option is turned on. If the mute option is on, the Mute button appears red on the IP Softphone 2050 interface.
- Check the NAT configuration. You receive one-way audio if the NAT configuration causes the IP Softphone 2050 to attempt to connect to an incorrect IP address.

### Broken or choppy speech

The Audio Quality slider adjusts the number of buffers between the computer audio device and the IP Softphone 2050 application. Less delay reduces the audio delay but increases the chances of getting broken or absent audio. High quality reduces the chances of broken audio but increases the audio delay.

If you receive choppy or broken speech, try moving the Audio Quality slider toward Higher Quality.

The following are other causes for audio quality issues on the IP Softphone 2050:

- Other applications that run on your computer, especially CPU-intensive applications or applications that intercept or delay packets (such as firewalls), can cause broken audio.
- Multiple Ethernet interfaces on a single PC may not work well together. For example, the IP Softphone 2050 can exhibit intermittent breaks in audio with notebooks that have a Xircom RealPort Cardbus Ethernet 10/100+Modem 56 PC Card (PCMCIA) inserted when the notebook is docked. In the case of this PC card, the problem relates to a version of the Xircom driver.
  This driver uses CPU cycles even when it is not active, which, in turn, causes IP Phone audio to become choppy. Solutions include upgrading your Xircom driver or removing the PC card when the notebook is docked. To upgrade your Xircom driver, download and install a new PC card driver from the Xircom Web site. You require

administrator privileges on Windows 2000; carefully follow the installation instructions provided by Xircom.

> ***Note:*** Removing the card without performing the Windows unplug or eject hardware procedure can cause the PC to reset.

## Audio commands

The following commands can be useful to debug audio-related problems.

- **`tsm_set_rx_gai`**
  Send a message to a DSP channel to set the Receive (RX) audio gain.
- **`tsm_set_tx_gai`**
  Send a message to a DSP channel to set the Transmit (TX) audio gain. Displays terminal data for each IP Phone.

The volume of the Receive (RX) and Transmit (TX) paths through the gateway is set through a combination of DSP pad levels and the fixed levels in the IP Phones. VGMC/Signaling Server extends the dynamic loss plan to the M1 system, so all system types use it. The Call Server sends the VGMC pad messages for each call to set the gains through the gateway. Use the commands in this section to examine the values used in this process.

The DSP level is specified as gain, so the in_gain and out_gain values printed by dimPrintChannelInfo are sign reversed from the normal loss values.

The Internet Phones and IP Softphone 2050 internally set unique gain levels internally to achieve the TIA-810A standard of RLR = +2 and SLR = +8. These values normally remain at the default levels, as these values are used in the loss plan setting for other areas of the system. However, some customers in the United Kingdom expect the louder levels (for example, non-TIA-810A/912 compliant) as offered by their existing digital phones. New CLI commands can be used to manually increase the IP Phone default levels.

- **`lossPlanClr`**
  Clear changes to the IP Phone default gain settings.
- **`lossPlanPrt`**
  Display the IP Phone gain settings.
- **`lossPlanSet`**
  Change the handset, headset, or handsfree gain settings by +/-8 dB.
- **`UKL`**
  Clear changes to the IP Phone default gain settings.

- **`UKLossPlanSet`**
  Increase the handset and headset transmit gain settings by 5 dB.
- **`usiGainTableShow`**
  Display the lookup table used to convert loss plan values in dB to the values the IP Phone uses for the CODEC and DSP control registers.
- **`usiQueryAPB`**
  Query the IP Phone for the current Audio Parameter Block data for transducer.
- **`usiShow`**
  Display terminal data for each IP Phone.

## VGMC and Signaling Server logging commands

Troubleshooting a problem can indicate a software bug that requires debugging. Your support team may ask to enable logging for specific tasks to collect additional information for a problem. You can use the following commands for this purpose:

- **`logConsoleOff`**, **`logConsoleOn`**
  Disable or enable printing of log messages to the VGMC serial port.
- **`logFileOff`**, **`logFileOn`**
  Disable or enable writing of log messages to the VGMC SYSLOG.n file.
- **`logPrintOff`**, **`logPrintOn`**
  Disable or enable printing of log messages on the VGMC to the active login session.
- **`logShow`**
  Display the current logging status for all tasks and the name of the current log file.
- **`rdxxxx`** commands (Signaling Server only)
  A set of commands to view the RPT log on the Signaling Server.
- **`syslogLevelSet`**
  Set the error level at which messages are printed for a task.
- **`syslogShow`**
  Display the current logging status for all tasks.

On the VGMCs, the SYSLOG function creates four files in the directory C:/LOG. These files are called SYSLOG.0, SYSLOG.1, SYSLOG.2, and SYSLOG.3 and are filled in a round-robin manner. The files are limited to 16 K to limit the logging function to 64 K of the C: drive disk space. The files allocate 16 K of disk space to limit fragmentation and disk corruption problems. Therefore, a file size of 16 K when `ll` is run does not necessarily mean the file has 16 K of data.

When a file reaches the 16 K limit, it is closed and the next file is opened. When OTM/EM retrieves the log file, a function on the card concatonates the four files in file date order and the resulting file is sent. EM retrieves and displays each file separately.

On the Signaling Server, the syslog output is written to the RPT log files in the /u/RPT directory. Each file is 1 MB in size. The contents of the RPT files are accessed using the RPT commands, the same commands used from the PDT shell of the CS 1000 (see "rdxxxx commands (Signaling Server only)" (page 223)).

## Voice Gateway trace commands

The following is a summary of the commands used to trace the voice Gateway.

**vgwTraceOn <chNum>, <vgw_trace_tool>**
Initiates the vgwTrace on a specified channel

**itgA07TraceOn <chNum>**
Initiates the tracing on the channel A07 messages

**vgwAudioTraceOn <chNum>**
Initiates the audio message tracing

**vgwRegistrationTraceOn <chNum>**
Initiates the tracing of registration messages for a specified channel

**usiLibTraceOn "IP_Addr", <LTPS ->Sets filter>,<Sets ->LTPS filter>**
Initiates the trace of the UNIStim messages

**tpsARTrace "type", "trace_id"**
Initiates the tpsAR protocol trace, which determines where a telephone must register.

## RTP and RTCP statistics

Three CLI commands, **`RTPStatShow`**, **`RTPTraceShow`** and **`RTPTraceStop`**, relate to real-time RTCP statistics. Upon issuing the CLI command RTPStatShow or RTPTraceShow to an IP Phone, the RTCP statistics report appears on the Signaling Server TTY port or Telnet session. RTPStatShow requests a snapshot of RTCP statistics versus RTPTraceShow requests RTCP statistics for a certain number of polling periods or until the end of the call. RTPTraceStop stops RTPTraceShow.

Both RTPStatShow and RTPTraceShow request an active IP Phone for RTCP statistics information. The only difference is that RTPStatShow requests a snapshot of RTCP statistics information but, RTPTraceShow periodically requests refreshed RTCP statistics information for a certain amount of time.

# VoIP

## Signaling Server base software

The Signaling Server base software provides functions such as logon service, event logging, patching, rlogin, Telnet, and object module loader.

### Signaling Server bootup

#### Initial BIOS screens

You require a full serial cable to see the BIOS boot screens. The hardware flow control is enabled during BIOS boot. The maintenance terminal emulates VT100 or ANSI. The default terminal speed is 19200 bps.

The following is a sample boot screen.

```
AMIBIOS (C)2001
American Megatrends Inc.
Copyright 1996-2001 Intel Corporation

TR440BXA.86B.0042.P15.0107200951

Intel(R) Pentium(R)III processor, 700MHz
256MB OK

Hit <F2> if you want to run SETUP
```

The BIOS version must be at least P15 (TR440BXA.86B.0042.P15.010720 0951). Some terminals do not pass function keys (such as F2 for SETUP). You may need to connect a PC keyboard directly to the Signaling Server.

#### Bootup sequence

The bootup sequence is as follows:

1. The BIOS is configured for bootup device order and to load boot track.
2. The boot track (that is, the boot loader or boot ROM) is read from floppy/CD/hard disk.

   a. Boot parameters are saved in the nvram.sys file.
   b. The nvram.sys file can exist on any boot device (floppy disk, CD, or hard disk).
   c. Changes to boot parameters are saved to the nvram.sys file (unless the device is read-only).
3. The boot ROM loads and starts the main load from floppy/CD/hard disk/network

As shown in the sample output, the character *c* reads the boot parameters from CD-ROM, and *h* reads the boot parameters from the hard disk. If a selection is not made, the menu times out and the default device [H] is selected.

This is required only if there is a bootable CD-ROM or boot floppy disk and a bootable hard disk. The purpose is to boot from hard disk when you have a bootable device in the drives.

```
Read boot parameters from:
[C]DROM
[H]ard Disk
3 [H]

Reading boot parameters from /p/nvram.sys
Press any key to stop auto-boot...
0
auto-booting...
```

If the auto boot menu stops if you unintentionally type some character, use the **@** command to continue the boot operation.

## Boot order

The following is boot order of the boot devices. A software boot automatically resets to this order:

1. ATAPI CDROM
2. IDE-HDD
3. Floppy

## General boot parameters

The boot parameters that are specified during the boot are:

- Boot device (interface)
- The main OS file to load (main OS module)
- ELAN network parameters (host name, IP, SM, GW)

- Network (FTP) server parameters (host name, IP)
- Boot options flag (leader, install, load all symbols, quick autoboot)
- Startup script (not normally used)

The following figure indicates the bits that constitute the boot flag.

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M3 | M2 | M1 | M0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | L3 | L2 | L1 | L0 |

M3 = 1, Do not load or run application programs

M3 = 0, Load application programs

M2 = 1, Not used

M1 = 1, Signaling Server is Leader

M1 = 0, Signaling Server is follower

M0 = 1, Install

L1 = 1, Load all symbols (including static/local/private symbols)

L1 = 0, do not load static, local, or private symbols

The leader boot parameters are:

- specify boot device: ata drive 0, controller 0
- specify file name (of mainos.sys, on hard disk)
- specify ELAN IP and SM
- specify ELAN GW
- specify flags=0x2000
- specify target name (my host name)
- specify other=fei (primary network interface (ELAN))

```
boot device : ata=0,0
unit number : 0
processor number : 0
file name : /p/mainos.sys
inet on Ethernet (e) : 47.11.216.XX:ffffff00
gateway inet (g) : 47.11.216.1
flags (f) : 0x2000
target name (tn) : my_ss
other (o) : fei
```

The follower boot parameters are:

- specify boot device: ata drive 0, controller 0
- specify file name (of mainos.sys, on hard disk)
- network parameters are obtained using BOOTP
- specify flags=0x0
- specify target name (my host name)
- specify no "other"

boot device : ata=0,0
unit number : 0
processor number : 0
file name : /cd0/mainos.sys
flags (f) : 0x0
target name (tn) : my_follower_ss

The CD-ROM installation boot parameters are:

- specify boot device: ata drive 0, controller 0
- specify file name (of mainos.sys, on CD-ROM)
- no network parameters
- specify flags=0x100a (installation)
- specify no "other"

```
boot device :  ata=0,0
unit number :  0
processor number :  0
file name :  /cd0/mainos.sys
flags (f) :  0x100a
```

## Signaling Server base commands

The following sections describe the commands you can use to retrieve information from the Signaling Server.

The commands are case-sensitive and the general convention is to provide the parameters for commands within double quotation marks.

The Telnet, rlogin and cslogin commands are available from OAM, PDT and VxWorks shell. You can use the **`cslogin`** command to rlogin from the Signaling Server to the Call Server and go directly to the overlay supervisor without knowing the CS PDT password.

## OAM shell commands

The commands available from OAM shell on the signaling Server are the commands provided by various applications. Type the command `help` to find the commands of various applications. The OAM shell is available upon a successful logon to the Signaling Server, represented as oam>.

The following system administration commands are available from the OAM shell.

**Table 3**
**Table 1: OAM shell commands**

| Command | Parameter | Description |
|---|---|---|
| telnet | [IP address or host name] | Telnet to the server specified by IP address or host name. |
| rlogin | [IP address or host name] | rlogin to the server specified by IP address or host name. |
| cslogin | | Logon to the Call Server overlays |
| routeShow | | Display host and network routing tables. |
| routeAdd | [destination][gateway] | Add a new route with destination and gateway to the routing tables. |
| arpShow | | Display the system ARP table |
| arpFlush | | Flush all entries in the system ARP table |
| swVersionShow | | Display the software version of the Signaling Server. |
| date | [] [Day Month Date hh:mm:ss Year] | Display date with no parameters, and configure date and time with specific parameters. |
| stty | [port speed] | Change the maintenance port speed. The acceptable values for the maintenance port speed are 9600, 19200, 38400 and 115200. |

**Table 3**
**Table 1: OAM shell commands (cont'd.)**

| Command | Parameter | Description |
|---|---|---|
| ppp | [-l local IP address] [-r remote IP address] [-o options file] | Set up a point-to-point connection from a terminal to the Signaling Server. Options available for debugging purposes are available in VxWorks documentation. |
| who | | Display the number (and who) of users connected to the Signaling Server. |

For information about the output of the preceding commands, see "OAM shell command reference" (page 70).

**Setting up a PPP connection to the Signaling Server** The following table illustrates a simple configuration to set a PPP connection to the Signaling Server.

**Table 4**
**PPP connection to the Signaling Server**

| Maintenance port on Signaling Server | Local IP address for Signaling Server | Remote IP address for Terminal |
|---|---|---|
| Back panel | 137.135.3.1 | 137.135.3.2 |
| Front panel | 137.135.5.1 | 137.135.5.2 |

The Signaling Server has two maintenance ports, one on the front face plate and the other back panel. The modem can connect to either maintenance port. The front maintenance port does not display system messages.

Set up the PPP connection from the Terminal by typing the command, ppp in a terminal window. The Signaling Server uses IP addresses assigned by default for this connection. The preceding table provides the default IP addresses that are assigned by the Signaling Server for the PPP session based on the maintenance ports used.

To use specific IP addresses to set up the PPP connection, issue the following command from the terminal window.

```
>ppp -l <Signaling Server-IPaddr> -r <Term-IPaddr>
```

Signaling Server-IPaddr is the IP address assigned to the Signaling Server and Term-IPaddr, the IP address assigned to the Terminal.

### Services switchover (OAM shell)

This sections shows all commands that can be used for services switchover.

`soHelpMenu`
Show commands you can use for Services Switch Over

`soCmdStatusShow`
Show service switch over commands status

**Graceful disable commands** This section contains the graceful disable commands for the IP Line, Virtual Trunk and Gatekeeper applications available from the OAM shell. These commands do not interrupt established calls. Graceful commands determine if available resources exist to reregister virtual trunks or IP Phones and cause the virtual trunks and telephones to reregister only when they are idle.

`disServices`
Cause the Voice Media Gateway Card (VGCM) or Signaling Server to gracefully switch the registered resources to the other VGMCs or Signaling Servers in the same node.

`disVTRK`
Cause the Signaling Server to gracefully switch the registered virtual trunks to another signaling Server in the same node.

`disTPS`
Cause the VGMC or Signaling Server to gracefully switch the registered line TPS to the other VGMCs or Signaling Servers in the same node.

`disGK`
Place the local gatekeeper out of service and the alternative gatekeeper in service.

**Force disable commands** This section contains the force disable commands for the IP Line, Virtual Trunk, and Gatekeeper applications available from the OAM shell. The force commands unregisters the virtual trunks and telephones, whether or not there is somewhere for these resources to reregister, which may strand the resources (for example, a telephone may have nowhere to reregister and continuously reboots, or virtual trunks may be disabled so you cannot make trunk calls with them). The force commands also cause active calls to be torn down (a trunk call using a virtual trunk resource is dropped, and a telephone on a call is reset.)

`forcedisServices`
Force all registered resources on the VGMC or Signaling Server to unregister and for gatekeeper to go out of service

`forcedisVTRK`
Force all registered virtual trunks to unregister from the local server

`forcedisTPS`
Force all telephones registered to the local line TPS to unregister

`forcedisGK`
Force the local gatekeeper to go out of service

**Enable commands** This section contains the enable commands for the IP Line, Virtual Trunk and Gatekeeper applications available from the OAM shell.

`enlServices`
Cause all VGMCs or Signaling Server to accept registrations of resources.

enlVTRK
Cause the Signaling Server to accept virtual trunk registrations. The virtual trunks are registered only to the master Signaling Server of the node. If the Signaling Server being enabled becomes the new master, the configured virtual trunks reregister to that Signaling Server.

`enlTPS`
Enable the TPS application and the TPS registration process on the system line TPS. Applies to both the VGMC and the signaling Server.

`enlGK`
Causes the local gatekeeper to go in service. The local gatekeeper will become active under the following conditions:

- It is configured as the Prime gatekeeper.
- It is configured as the Alternative gatekeeper, and the Prime gatekeeper is out of service
- It is configured as the Fail Safe gatekeeper and both the Prime gatekeeper and the Alternative gatekeeper are out of service.

The local gatekeeper will go into standby when:

- It is configured as the Alternative gatekeeper and the Prime gatekeeper goes back in service (active)
- It is configured as Fail Safe and the Prime gatekeeper or the Alternative gatekeeper goes back in service (active)

**Other commands** This section contains other useful commands available from the OAM shell.

loadBalance
Cause the VGMC or Signaling Server to attempt to balance the registration load of telephones between this card or server and the remaining node components. If a Signaling Server is in the node, no SMC or ITG-P take over any telephones. This command works only within a group of servers of the same type (Signaling Servers, ITG-Ps, SMCs)

`servicesStatusShow`
Show the status of all services configured in the local platform.

## Trace and diagnostic commands

### H323 Trace Commands

This section contains OAM commands useful for tracing H323 messaging for active trunk calls.

`H323CallTrace <on or off>`
Trace all incoming and outgoing messages for all channels

`H323CallTrace <on or off(MsgRecv)><on or off (MsgSend)>`
Trace the incoming or outgoing messages for all channels.

`H323CallTrace <channel #><on or off (MsgRecv)><on or off (MsgSend)>`
Trace the incoming or outgoing messages for a specific channel.

`H323CallTrace <start channel #><end channel #><on or off (MsgRecv)><on or off (MsgSend)>`
Trace the incoming or outgoing messages for a range of channels

`H323Output <1(tty) or 2(rpt)><on or off>`
Direct the H323CallTrace output to the TTY or to the RPT.LOG

`H323TraceShow`
Show the input and output display settings for H323CallTrace and the H323Output settings.

### DCH diagnostic tool

The `DCHmenu` command is available from the OAM shell on the Signaling Server.

`DCHmenu`
Display a menu of DCH diagnostic tools

### Maintenance terminal

You can configure the maintenance terminal port speed. This change is saved to BIOS, so that it survives reboots and power cycles.

**stty <speed>**

Set port speed on the maintenance ports on the front and back of the Signaling Server. Available speeds are 9600, 19200, 38400 and 11520.

### OAM shell command reference

The section contains sample output from OAM shell commands. The commands are listed alphabetically. For more information about commands not listed here, see "VoIP command reference" (page 82) or *Signaling Server OAM Shell Command Line Reference,* (NN48500-543).

```
oam> date
MON OCT 28 18:00:48 2002
oam> date Tue OCT 29 10:00:00 2002
oam> date
TUE OCT 29 10:00:01 2002

oam> DCHmenu
Please select one of the DCHmenu options:

0 - Print menu (default)
1 - Print current DCH state
2 - Print current DCH configuration
3 - Print application error log
4 - Print link error log
5 - Print protocol error log
6 - Print message log
7 - Enable printing all messages processed by UIPC
8 - Enable error printing
9 - Enable info printing
10 - Enter manual message mode
11 - Print b channel control blocks
99 - Exit menu

Please enter your DCHmenu choice (0 to print the menu):

oam> disGK
oam> 03/12/03 03:32:47 LOG0006 tSOGK: disGK: Puts out
of service the local gatekeeper and puts in service the
alternative gatekeeper if available 03/12/03 03:32:47
LOG0006 tSOGK: The disGK command has failed.  Reason:
Unreachable Alternative GK. Available command:  forcedisGK

oam> disServices
oam> 03/12/03 03:36:57 LOG0006 VTRK: The disVTRK command
has failed.  Reason:  There are not enough virtual
trunk resources in the node to handle reregistration of
resources.  Available command:  forcedisVTRK
03/12/03 03:36:58 LOG0006 tSO: disGK: Puts out of service
the local gatekeeper and puts in service the alternative
gatekeeper if available
03/12/03 03:36:58 LOG0006 tSO: The disGK command has
failed.  Reason:  Unreachable Alternative GK. Available
command:  forcedisGK
03/12/03 03:36:58 LOG0006 TPS: Virtual trunk application
is active.  First disable virtual trunks and then reissue
disTPS command

oam> disTPS
oam> 03/12/03 03:32:06 LOG0006 TPS: Virtual trunk
application is active.  First disable virtual trunks and
then reissue disTPS command

oam> disVTRK
oam> 03/12/03 03:27:13 LOG0006 VTRK: The disVTRK command
has failed.  Reason:  There are not enough virtual
trunk resources in the node to handle reregistration of
resources.  Available command:  forcedisVTRK

oam> enlGK
oam> 03/12/03 03:25:29 LOG0006 tSOGK: enlGK: GK is already
in service

oam> enlServices
oam> 05/12/03 10:02:11 LOG0006 shell:  Causes the vitrual
trunk application to be enabled and to accept virtual trunk
registrations
05/12/03 10:02:11 LOG0006 shell:  vtrkStateHandler:
VtrkSOEnable:  No state change (state = VtrkStandby)
05/12/03 10:02:11 LOG0006 shell:  enlGK: Causes the local
gatekeeper to be put in service
05/12/03 10:02:11 LOG0006 GKNPM: In Service:  Switching to
GK_STANDBY
05/12/03 10:02:11 LOG0006 shell:  GK in service
05/12/03 10:02:11 LOG0006 TPS: Service enabled
05/12/03 10:02:11 LOG0006 SET: Service enabled
:
:

oam> enlTPS
oam> 03/12/03 03:38:16 LOG0006 shell:  LTPS is enabled,
enlTPS was ignored

oam> enlVTRK
oam> 03/12/03 03:44:44 LOG0006 shell:  Causes the vitrual
trunk application to be enabled and to accept virtual trunk
registrations
03/12/03 03:44:44 LOG0006 shell:  vtrkStateHandler:
VtrkSOEnable:  No state change (state = VtrkStandby)
03/12/03 03:44:46 LOG0006 GKNPM: gkNpmHandleRRQRequest:
GK_OUT_OF_SERVICE
03/12/03 03:44:46 LOG0005 GKNPM: RAS FAILURE: RAS_TYPE :
RRQ,Reason=discoveryRequired,SrcIP=47.11.217.205:1719
03/12/03 03:44:59 LOG0003 tNetTask:  [ARP] duplicate
IP address 2f0bf98b sent from ethernet address
00:60:38:bd:06:31
03/12/03 03:44:59 LOG0006 CSV: CSV enable
03/12/03 03:44:59 LOG0006 CSV: Node 1999 registering for
terminal connections on 47.11.249.139:4100
03/12/03 03:44:59 LOG0006 TPS: No security checking for
this card
03/12/03 03:44:59 LOG0004 UMS: No firmware for i2001 was
found in /u/fw/
03/12/03 03:44:59 LOG0006 ELC: VTRK: This signal server is
master
03/12/03 03:44:59 LOG0006 ELC: gkNpmCardEventHandler:
unhandled event 0x4
03/12/03 03:44:59 LOG0006 VTRK: vtrkStateHandler:
VtrkElecWon:  State change from VtrkStandby to
VtrkRegistration
03/12/03 03:44:59 LOG0006 VTRK: server SSRC is 2048
03/12/03 03:44:59 LOG0006 VTRK: leaderFlag is 16777216
03/12/03 03:44:59 LOG0006 VTRK: after htonl ssrc is 524288
03/12/03 03:44:59 LOG0006 VTRK: server <LeadeSS> node
<1999> online announce
03/12/03 03:44:59 LOG0006 NPM: npmMasterUpdate:  register
with GK
03/12/03 03:44:59 LOG0005 ELC: Election won, master =
47.11.249.176
03/12/03 03:44:59 LOG0006 GKNPM: gkNpmHandleRRQRequest:
GK_OUT_OF_SERVICE
03/12/03 03:44:59 LOG0005 GKNPM: RAS FAILURE: RAS_TYPE :
RRQ,Reason=discoveryRequired,SrcIP=47.11.249.139:1719
03/12/03 03:44:59 LOG0003 NPM: cmEvRASReject:  GK
47.11.249.176 Registration rejected
03/12/03 03:44:59 LOG0003 NPM: cmEvRASReject:  GK
47.11.215.243 Registration rejected
03/12/03 03:44:59 LOG0003 VTRK: itgMsgSend to task 0xf800
```

```
03/12/03 03:44:59 LOG0003 VTRK: vtrkCDSInfoHandler:  send
CDS Info to SIPNPM failed
03/12/03 03:44:59 LOG0003 VTRK: itgMsgSend to task 0xf800
03/12/03 03:44:59 LOG0003 VTRK: vtrkCDSInfoHandler:  send
CDS Info to SIPNPM failed
03/12/03 03:44:59 LOG0006 VTRK: ServerStatus OK
03/12/03 03:44:59 LOG0006 VTRK: vtrkStateHandler:
VtrkCSRegOK: State change from VtrkRegistration to
VtrkActive
03/12/03 03:44:59 LOG0004 VTRK: DCH Established

oam> forcedisGK
oam> 03/12/03 03:40:28 LOG0006 tSOGK: forcedisGK: Forces
the local gatekeeper to be put out of service
03/12/03 03:40:28 LOG0004 tSOGK: gkNpmResultCheck:
unhandle gkNpmMsgType 11 for OOS_NO_ALT_GK_CONFIG
03/12/03 03:40:28 LOG0006 GKNPM: OOS Force:  Switching to
GK_OUT_OF_SERVICE
03/12/03 03:40:29 LOG0006 GKNPM: gkNpmHandleRRQRequest:
GK_OUT_OF_SERVICE

oam> forcedisServices
oam> 05/12/03 09:55:39 LOG0006 shell:  Forces all
registered virtual trunks to unregister from the local
server
05/12/03 09:55:39 LOG0006 shell:  vtrkStateHandler:
VtrkSOForceDisable:  State change from VtrkActive to
VtrkDeregistration
05/12/03 09:55:39 LOG0006 shell:  VTRK Unregister all
05/12/03 09:55:39 LOG0006 shell:  server <S_Campbell_SS>of
fline announce
05/12/03 09:55:39 LOG0006 shell:  forcedisGK: Forces the
local gatekeeper to be put out of service
05/12/03 09:55:39 LOG0003 GKNPM: OOS Force:  Switching to
GK_OUT_OF_SERVICE
05/12/03 09:55:39 LOG0006 shell:  GK out of service
:
:

oam> forcedisTPS
oam> 03/12/03 03:43:44 LOG0006 TPS: Force disable TPS
03/12/03 03:43:44 LOG0006 SET: Service force disabled,
Reset All Sets

oam> forcedisVTRK
oam> 03/12/03 03:41:21 LOG0006 shell:  Forces all
registered virtual trunks to unregister from the local
```

```
server
03/12/03 03:41:21 LOG0006 shell:  vtrkStateHandler:
VtrkSOForceDisable:  State change from VtrkActive to
VtrkDeregistration
03/12/03 03:41:21 LOG0006 shell:  VTRK Unregister all
03/12/03 03:41:21 LOG0006 shell:  server <LeadeSS>offline
announce
03/12/03 03:41:21 LOG0003 VTRK: ITG2106 DCH 15 Failure (91)
03/12/03 03:41:21 LOG0006 VTRK: vtrkStateHandler:
VtrkLastTNUnreg:  State change from VtrkDeregistration to
VtrkStandby
03/12/03 03:41:21 LOG0006 CSV: CSV disabled
03/12/03 03:41:21 LOG0006 ELC: VTRK: This signal server is
not a master
03/12/03 03:41:21 LOG0006 ELC: gkNpmCardEventHandler:
unhandled event 0x4 03/12/03 03:41:21 LOG0004 VTRK:
vtrkStateHandler:  event 1 ignored ; state = VtrkStandby
03/12/03 03:41:21 LOG0006 NPM: npmMasterUpdate:
registration timer killed
03/12/03 03:41:21 LOG0006 NPM: npmMasterUpdate:  already
unregistered
03/12/03 03:41:21 LOG0003 VTRK: itgMsgSend to task 0xf800
03/12/03 03:41:21 LOG0003 VTRK: vtrkCDSInfoHandler:  send
CDS Info to SIPNPM failed
03/12/03 03:41:21 LOG0003 VTRK: itgMsgSend to task 0xf800
03/12/03 03:41:21 LOG0003 VTRK: vtrkCDSInfoHandler:  send
CDS Info to SIPNPM failed 03/12/03 03:41:21 LOG0006 VTRK:
ServerStatus OK
03/12/03 03:41:21 LOG0004 VTRK: vtrkStateHandler:  event 6
ignored ; state = VtrkStandby

oam> H323CallTrace

H323CallTrace

Usage:  A) H323CallTrace <on or off>
B) H323CallTrace <on or off (MsgRecv)><on or off (MsgSend)>
C) H323CallTrace <channel #><on or off (MsgRecv)><on or off
(MsgSend)>
D) H323CallTrace <start channel #><end channel #><on or off
(MsgRecv)><on or off (MsgSend)>

oam> H323TraceShow
TTY output:  OFF
RPT output:  ON

Channels H323MsgRecv (VTRK->NPM) H323MsgSend (NPM->VTRK)
```

```
======== ====================== ======================
0 - 382 ON ON

VGMC> loadBalance
value = 0 = 0x0
VGMC>
DEC 05 09:43:49 tShell:  Info Line TPS is attempting to
balance the registration load of sets between this card and
the rest of the node components DEC 05 09:43:54 TPS: Info
Average number of sets registered to a signaling server
should be:  0.  Average number of sets registered to a VGMC
should be:  3
DEC 05 09:43:54 SET: Info Load balance message received from
TPS
oam> rlogin 47.11.255.29

login:
password:

oam> routeAdd 47.11.255.0 47.11.255.30
oam> routeShow

ROUTE NET TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
---------------------
0.0.0.0 47.11.249.1 3 0 88 fei1
47.11.249.0 47.11.249.111 101 0 0 fei1
47.11.254.0 47.11.255.29 101 0 0 fei0
-------------------------------------------------------
---------------------

ROUTE HOST TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
---------------------
47.11.255.0 47.11.255.30 7 0 0 fei0
127.0.0.1 127.0.0.1 5 1 1 lo0
-------------------------------------------------------
---------------------
oam> routeShow

ROUTE NET TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
---------------------
0.0.0.0 47.11.249.1 3 0 84 fei1
```

```
47.11.249.0 47.11.249.111 101 0 0 fei1
47.11.254.0 47.11.255.29 101 0 0 fei0
-------------------------------------------------------
---------------------
ROUTE HOST TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
---------------------
127.0.0.1 127.0.0.1 5 1 1 lo0
-------------------------------------------------------
---------------------

oam> soCmdStatusShow

VTRK Services Switch Over Command Status Show
---------------------------------------------
Command:  forcedisVTRK
Status:  Successful
Reason:  Forces all registered virtual trunks to unregister
from the local server.
All virtual trunk TNs will be reset within a couple of
minutes

GK Services Switch Over Command Status Show
-------------------------------------------
Command:  forcedisGK
Status:  Successful
Reason:  Forces the local gatekeeper to be put out of
service

LTPS Services Switch Over Command Status Show
---------------------------------------------
Command:  forcedisTPS
Status:  Successful
Reason:  Forces all registered line LTPS to unregister from
the local server.
All sets will be reset within a couple of minute

oam> soHelpMenu
Services Switch Over Help Menu
------------------------------
Graceful disable services
disServices:  Causes the server to gracefully switch the
registered resources to the other services in the same node
disTPS: Causes the line LTPS to gracefully switch the
registered sets to the other cards located in the same node
disVTRK: Causes the virtual trunk to gracefully switch the
```

```
registered virtual trunks to other SS located in the same
node
disGK: Puts out of service the local gatekeeper and puts
in service the alternative gatekeeper if available Force
disable services
forcedisServices:  Forces the server to switch the
registered resources to the other services in the same node
forcedisTPS: Forces all registered line LTPS to unregister
from the local server
forcedisVTRK: Forces all registered virtual trunks to
unregister from the local server
forcedisGK: Forces the local gatekeeper to be put out of
service Enable services
enlServices:  Causes all the services to accept
registration of resources
enlTPS: Causes line LTPS application to be enabled and to
accept set registrations
enlVTRK: Causes the vitrual trunk application to be enabled
and to accept virtual trunk registrations enlGK: Causes the
local gatekeeper to be put in service
loadBalance:  Causes the service to attempt to balance the
registration load of sets between this service and the rest
of the node services
servicesStatusShow:  Shows the status of services
(tps/iset/vtrk/gk)
soCmdStatusShow:  Shows the service switch over commands
status
oam>

oam> swVersionShow
sse-2.00.74 Wednesday October 16 2002 20:04:18 EDT

Loaded Modules:
share.obj sse-2.00.74
line.obj sse-2.00.74
trunk.obj sse-2.00.74
gk.obj sse-2.00.74
web.obj sse-2.00.74

oam>stty 9600
oam>

oam> telnet 47.11.255.29
Trying 47.11.255.29 ....

login:
password:
```

```
oam> who
3 /tyCo/0 nobody
5 /tyCo/1 target
42 /pty/pty00.S target 47.11.181.81
->device id->device name->username->connected IP(if
applicable)]
```

## PDT shell commands

To enter the PDT shell, type CTRL+p+d+t. The Patching facility commands are available from the PDT shell (pdt>). Type **`help Patcher`**.

**Table 5**
**Patch commands**

| Command | Parameter | Description |
|---|---|---|
| pload | [patch filename] | Load patch into memory |
| pins | patch handle | Place a patch in service |
| poos | patch handle | Remove a patch from service |
| pout | patch handle | Remove a patch from memory |
| plis | patch handle | List details of a specific patch |
| pstat | [patch handle] | List status of all active patches |
| pnew | patch filename | Create memory patches |

If the these commands are run from the vxshell, you receive an error.

The software patch file is downloaded from a workstation to the Signaling Server. The patch files are stored in flash memory. Load the patch file into DRAM memory by using the **pload** command.

In the PDT shell, typing help system lists the following commands. Additionally, all OAM Shell commands are available in the PDT shell.

**Table 6**
**PDT system commands**

| Command | Parameter | Description |
|---|---|---|
| devs | – | Show the list of the devices |
| echo | – | Echo the inputs |
| hosts | – | Display hosts list |
| memShow | – | Display memory usage |
| ti | [taskname or taskid] | Display task information |
| i | [taskname or taskid] | Display task information |
| version | – | Display VxWorks version |

**Table 6**
**PDT system commands (cont'd.)**

| Command | Parameter | Description |
|---|---|---|
| who | – | Display all active rlogin user ID and ports |
| x | [functionname] | Execute a function |
| ifShow | [networkinterfacename][] | Display the attached network interfaces. Display all network interfaces with no parameters. |
| reboot | [][-1] | Perform a warm restart with no parameters. Perform a cold restart with the value –1. |
| ls | ["path"[,long]] | List the contents of a directory |
| ll | ["path"] | Do a long listing of a directory contents |
| pwd | – | Print the current default directory |
| cd | "path" | Change the default directory |
| remove | "path" | Remove a file |
| copy | ["in"][,"out"] | Copy from input file to output file |
| rename | "old","new" | Rename or move one file to another |
| moduleShow | – | Show the list of all loaded modules |
| inetstatShow | – | Display all active connection for the IP sockets |
| tcpstatShow | – | Display statistics for the TCP protocol |
| udpstatShow | – | Display statistics for the UDP protocol |
| syslogShow | – | Display the log level for all tasks |
| syslogLevelSet | [tid\|name level] | Configure the log level for a task, given by task ID, or task name. The level is a number in the range of 0 to 7. |

Type **help rdtools** to obtain the commands to display information from the report log file.

The report log files are stored in the directory /u/rpt with file name in the format LOG000nn.RPT, where nn are numbers. The higher the nn number, the more recent report logs.

**Table 7**
**Report log file commands**

| Command | Parameter | Description |
|---|---|---|
| rdopen | [filename] | Open a report log file |
| rdgo | [N] | Go to a specific record, where N is absolute record number. |
| rd | [S][R] | Display records. Go S steps and display R records. Both S and R may be positive or negative. |
| rds | [S][R] | Display records with symbolic dump. Go S steps and display R records. |
| rdshow | – | Show general log file information |
| rdall | – | Display all records, without symbolic dump. |
| rdtail | [N] | Display N newest records, without symbolic dump. |
| rdhead | [N] | Display oldest N records, without symbolic dump. |
| rdnext | – | Open the next log file in the list of generated log files |
| rdprev | – | Open the previous log file in the list of generated log files |

The rdopen command issued with no parameters opens the most recent report log file. To open a particular log file, enter the report log file name as the parameter for the rdopen command.

The following example shows output from report log file commands.

```
pdt> rdopen "/u/rpt/LOG00009.RPT"
Reading /u/rpt/LOG00009.RPT

pdt> rdgo 230
[0230] 28/10/02 15:20:00 LOG0006 NPM: npmControlTOSGet:
H323 Control Layer3 TOS is:  0x28

pdt> rd 0, 5
[0230] 28/10/02 15:20:00 LOG0006 NPM: npmControlTOSGet:
```

```
H323 Control Layer3 TOS is:  0x28
[0231] 28/10/02 15:20:00 LOG0004 NPM: npmH323Init:  not
master, abort
[0232] 28/10/02 15:20:01 LOG0006 tRootTask:  Task npmInit
initialization succeeded
[0233] 28/10/02 15:20:01 LOG0006 NPM: tNpm task init
successful
[0234] 28/10/02 15:20:03 LOG0006 HTTP: SYSLOG initialised
pdt> rds 0, 5
[0235] 28/10/02 15:20:03 LOG0006 HTTP: Memory file system
initialised!
[0236] 28/10/02 15:20:03 LOG0006 HTTP: Setup HTTP Aliasing
[0237] 28/10/02 15:20:03 LOG0006 HTTP: Setup HTTP File
System
[0238] 28/10/02 15:20:03 LOG0006 HTTP: Setup Server Side
Includes
[0239] 28/10/02 15:20:03 LOG0006 HTTP: Load web server
config file successful!

pdt> rds 10, 10
[0250] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  7 edd
[0251] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  8 err
[0252] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  9 esn
[0253] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  10 hwr
[0254] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  11 ini
[0255] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  12 itg
[0256] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  13 npr
[0257] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  14 ovl
[0258] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  15 pri
[0259] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  16 rpt

pdt> rdtail 5
[269] 28/10/02 15:20:04 LOG0006 HTTP: Task httpd
initialization succeeded
[268] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  25 tfc
[267] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
```

```
Error Look up table index file:  24 mph
[266] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  23 misp
[265] 28/10/02 15:20:04 LOG0006 HTTP: Finish loading the
Error Look up table index file:  22 msdl

pdt> rdhead 5
[0000] 28/10/02 14:50:21 LOG0006 tRootTask:  alarmInit
initialization succeeded
[0001] 28/10/02 14:50:21 LOG0006 tRootTask:  ITG5000 Card
initialized, all alarms cleared.  (202)
[0002] 28/10/02 14:50:21 LOG0006 tRootTask:  shareAnnounce
initialization succeeded
[0003] 28/10/02 14:50:21 LOG0006 tRootTask:  ELAN IP =
47.11.255.29
[0004] 28/10/02 14:50:21 LOG0006 tRootTask:  itgCardInit
initialization succeeded

pdt> rdnext
Reading /u/rpt/LOG00002.RPT

pdt> rdprev
Reading /u/rpt/LOG00001.RPT
```

## VoIP command reference

The commands and variables in this section are listed alphabetically. For more information about the VoIP commands, see *Signaling Server OAM Shell Command Line Reference,* (NN48500-543).

### activeDlogShow

Syntax:

**`activeDlogShow numOfLine`**

Use this command to show the active log file information for the UFTP IP Telephone firmware download.

This command does not apply to MGC, MC32S, VGMC, or the Signaling Server.

The following table describes the command parameters.

**Table 8**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| numOfLine | – | Optional. Specify the number of lines to print.<br><br>When no argument is passed, the command shows the contents of the entire file. |

The following example shows command output.

```
oam> activeDlogShow
Active F/W download file:  /u/log/UFTPLOG0.TXT
Space remaining:  55

/u/log/UFTPLOG0.TXT
----------------------------------------
12/29/03 19:58:41 f/w dnld success:  (47.11.217.11) I2004
12/29/03 20:24:30 f/w dnld success:  (47.11.217.12) I2004
12/29/03 21:42:11 f/w dnld success:  (47.11.217.15) I2002
12/29/03 22:17:40 f/w dnld success:  (47.11.217.20) I2004
```

### arpFlush

Syntax:
**`arpFlush`**

Flushes all nonpermanent entries from the card ARP cache.

The following example shows output from the ITG-P card.

```
-> arpShow
LINK LEVEL ARP TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
-------------------
192.168.1.10 00:00:75:45:38:aa 405 1 1081508 lnIsa0
192.168.1.14 00:60:38:01:09:fa 405 2 1836 lo0
192.168.1.102 00:c0:4f:ae:a8:39 405 1 3 lnIsa0
192.168.1.140 00:60:38:01:a1:46 405 0 6 lo0
192.168.1.141 00:60:38:76:21:af 405 0 201 lnPci1
192.168.1.142 00:60:38:76:21:a9 405 3 136956 lnPci1
-------------------------------------------------------
-------------------
value = 75 = 0x4b = 'K'
->
-> arpFlush
value = 0 = 0x0
-> arpShow
```

```
LINK LEVEL ARP TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
-------------------
192.168.1.14 00:60:38:01:09:fa 405 2 1936 lo0
192.168.1.140 00:60:38:01:a1:46 405 0 6 lo0
-------------------------------------------------------
-------------------
value = 75 = 0x4b = 'K'
```

The following example shows output on the SMC card.

```
-> arpShow

LINK LEVEL ARP TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
------------------- 47.11.215.1 00:e0:16:78:5b:83 405 1 0
ixpMac0
47.11.215.44 00:03:47:da:cd:5a 405 0 312 ixpMac0
47.11.215.159 00:60:38:bd:20:92 405 1 312 ixpMac0
47.11.216.180 00:00:75:45:1e:d 405 2 2454 ixpMac1
47.11.217.158 00:03:47:da:cd:59 405 1 28 ixpMac1
-------------------------------------------------------
-------------------
value = 75 = 0x4b = 'K'
-> arpFlush
value = 0 = 0x0
-> arpShow

LINK LEVEL ARP TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
-------------------
47.11.215.1 00:e0:16:78:5b:83 405 1 0 ixpMac0
-------------------------------------------------------
-------------------
value = 75 = 0x4b = 'K'
```

The following example shows output on the Signaling Server.

```
-> arpShow

LINK LEVEL ARP TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
```

```
-------------------
47.11.215.1 00:e0:16:78:5b:83 405 1 0 fei1
47.11.215.30 00:60:38:bd:b3:50 405 0 343 fei1
47.11.215.44 00:03:47:da:cd:5a 405 3 1014 lo0
47.11.215.46 00:60:38:76:e9:85 405 0 1241 fei1
47.11.215.47 00:60:38:76:e9:75 405 2 2629 fei1
47.11.215.159 00:60:38:bd:20:92 405 0 361 fei1
47.11.215.161 00:60:38:76:e9:9f 405 1 2706 fei1
47.11.215.163 00:60:38:76:a5:f8 405 0 1030 fei1
47.11.216.1 00:e0:16:78:5b:84 405 0 0 fei0
47.11.216.180 00:00:75:45:1e:d 405 2 59129 fei0
47.11.216.181 00:60:38:8e:29:b9 405 0 31 fei0
47.11.216.246 00:60:38:bd:b3:51 c05 1 39 fei0
-------------------------------------------------------
-------------------
value = 75 = 0x4b = 'K'
-> arpFlush
value = 0 = 0x0
-> arpShow

LINK LEVEL ARP TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
-------------------
47.11.215.1 00:e0:16:78:5b:83 405 1 0 fei1
47.11.215.44 00:03:47:da:cd:5a 405 2 1014 lo0
-------------------------------------------------------
-------------------
value = 75 = 0x4b = 'K'
```

## arpShow

Syntax:

**`arpShow`**

Display the current entries in card system ARP table. See “arpFlush” (page 83) for an example.

## auditReboot

Syntax:

**`auditReboot value`**

This command specifies whether the VGMC audit task reboots the card when it detects a suspended task.

The following table describes the command parameters.

**Table 9**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| value | 0, 1 | 0 = Do not reboot the card<br>1 = Reboot the card |

The following example shows command output.

```
VGMC>auditReboot 1
Reboot when detect a suspended task --- Enabled value = 8 =
0x8
VGMC> auditReboot 0
Reboot when detect a suspended task --- Disabled value = 9 =
0x9
```

## auditShow

Syntax:
**`auditShow`**

Displays the current setting for the action to take upon detection of a suspended task. Enabled means that the VGMC reboots if a suspended task is detected. Disabled means that the VGMC does not reboot. The output also lists the reboot time (in 24-hour format) and lists critical and noncritical tasks.

The following example shows command output.

```
VGMC> auditShow
Reboot when detect a suspended task --- Enabled
Reboot Time :  2:00
Critical Task :  tTPS tVTM tSET tVTI tUMS tUMC tRDP tVGW tRTP
tRTCP tELC baseMMintTask tA07 tShell tNetTask tExcTask
tTelnetd
Non-Critical Task:  tLogTask tAioWait tAioIoTask1
tAioIoTask0 tPcmciad tTffsPTask tPortmapd tRdbTask
tFtpdTask tTftpdTask tSnmpd tITGStart tSyslogd tPxTimer
tbootpd tSntpsTask tXA tMAM tMVX_XSPY tMVX_DIM tOMM
tRPCMGMT tCSV tTPSAR tfwBk midnightTask tTelnetOutTask
tTelnetInTask tyLstnr tPingTmo0 tPingTx0 tREQ tPingTmo1
tPingTx1 tFtpdServ1 tFtpdServ2 tPingTmo2 tPingTx2
tPingTmo3 tPingTx3 tPingTmo4 tPingTx4 tPingTmo5 tPingTx5
tPingTmo6 tPingTx6 tPingTmo7 tPingTx7 tPingTmo8 tPingTx8
tPingTmo9 tPingTx9 tPingTmo10 tPingTx10 tPingTmo11
tPingTx11 tPingTmo12 tPingTx12 tPingTmo13 tPingTx13
tPingTmo14 tPingTx14 tPingTmo15 tPingTx15 tPingTmo16
tPingTx16 tPingTmo17 tPingTx17 tPingTmo18 tPingTx18
tPingTmo19 tPingTx19 tPingTmo20 tPingTx20 tPingTmo21
```

```
tPingTx21 tPingTmo22 tPingTx22 tPingTmo23 tPingTx23
tPingTmo24 tPingTx24 tPingTmo25 tPingTx25 tPingTmo26
tPingTx26 tPingTmo27 tPingTx27 tPingTmo28 tPingTx28
tPingTmo29 tPingTx29 tPingTmo30 tPingTx30 tPingTmo31
tPingTx31 tPingTmo32 tPingTx32 tPingTmo33 tPingTx33
tPingTmo34 tPingTx34 tPingTmo35 tPingTx35 tPingTmo36
tPingTx36 tpbxResetMain

value = 1083 = 0x43b
VGMC>
```

### bootpdDump

Syntax:

**`bootpdDump`**

Display the current BOOTP server database.

The following example shows command output.

```
-> bootpdDump
value = 0 = 0x0
->
# main 2.4.3
# (null):  dump of bootp server database.
# Dump taken SUN MAY 26 20:43:36 1996
#
# Legend:  (see bootptab.5)
# first field -- hostname (not indented)
# bf -- bootfile
# bs -- bootfile size in 512-octet blocks
# cs -- cookie servers
# df -- dump file name
# dn -- domain name
# ds -- domain name servers
# ef -- extension file
# ex -- exec file (YORK_EX_OPTION)
# gw -- gateways
# ha -- hardware address
# hd -- home directory for bootfiles
# hn -- host name set for client
# ht -- hardware type
# im -- impress servers
# ip -- host IP address
# lg -- log servers
# lp -- LPR servers
# ms -- message size
# mw -- min wait (secs)
# ns -- IEN-116 name servers
# nt -- NTP servers (RFC 1129)
# ra -- reply address override
# rl -- resource location protocol servers
# rp -- root path
# sa -- boot server address
# sm -- subnet mask
# sw -- swap server
# tc -- template host (points to similar host entry)
# td -- TFTP directory
# to -- time offset (seconds)
# ts -- time servers
# vm -- vendor magic number
# yd -- YP (NIS) domain
# ys -- YP (NIS) servers
# Tn -- generic option tag n

1:\
:dn=003 0:\
:gw=192.168.1.1:\
:hn:\
:ht=1:ha="00:60:38:01:09:FA":\
:ip=192.168.1.14:\
:lp=192.168.1.140, 255.255.255.192, 192.168.1.200:\
:sm=255.255.255.192:\
:to=0:\
:ts=192.168.1.149:

2:\
:dn=007 0:\
:gw=192.168.1.1:\
:hn:\
:ht=1:ha="00:60:38:01:19:4F":\
:ip=192.168.1.15:\
:lp=192.168.1.150, 255.255.255.192, 192.168.1.200:\
:sm=255.255.255.192:\
:to=0:\
:ts=192.168.1.149:
.subnet1:\
:gw=192.168.1.1:\
:hn:\
:sm=255.255.255.192:\
:ts=192.168.1.149:

->
```

## bootpdReload

Syntax:

**`bootpdReload`**

Reload and parse the BOOTP.TAB file.

The following example shows command output.

```
-> bootpdReload
value = 0 = 0x0
->
```

## bootPFileGet

Syntax:

**`bootPFileGet "srvrIP","uid","pswd","path","filename"`**

This command loads the file from the specified FTP server, renames it as bootp.tab, and writes it to the CONFIG directory. If the card is the Leader, a copy of the existing BOOTP.TAB file called BOOTP.BAK is created first. Also, if the card is the Leader, the new file is transferred using FTP to the other cards in the node. Downloading the node properties from OTM triggers the same behaviour.

The path and file name parameters specify the path and file on the FTP server. For examples using user IDs and passwords for various FTP servers, see "swDownload " (page 258).

The following example shows command output.

```
VGMC> bootPFileGet "192.168.1.15","itgadmin","itgadmin",
"/C:/","BOOTP.1"
value = 0 = 0x0
Checking card at f01a8c0
Doing card at f01a8c0

FEB 15 12:45:51 tbootpdSync:  Error Cannot set lock for file
type 4
FEB 15 12:45:51 tbootpdSync:  Notice File transfer
starting:  /C:/config/bootp.tab -> 192.168.1.15:/C:/CONFI
G/BOOTP.TAB
FEB 15 12:45:51 tbootpdSync:  Notice File transfer
completed:  /C:/config/bootp.ta b -> 192.168.1.15:/C:/CON
FIG/BOOTP.TABChecking card at 0
```

## bootPFilePut

Syntax:

**`bootPFilePut "hostIP","uid","pswd","path","fname"`**

This command loads the BOOTP.TAB from the card /CONFIG directory to the specified host. The path and fname parameters specify the directory on the remote host and the name of the file to create. For examples using user names and passwords for various FTP servers, see "swDownload " (page 258).

The following example shows command output.

```
VGMC> bootPFilePut "192.168.1.15","itgadmin","itgadmin",
"/C:/","BOOTP.1"
value = 0 = 0x0
VGMC>
FEB 15 12:44:22 tfx4:  Notice File transfer starting:
/C:/config/bootp.tab -> 192.168.1.15:/C://BOOTP.1
FEB 15 12:44:23 tfx4:  Notice File transfer completed:
/C:/config/bootp.tab -> 192.168.1.15:/C://BOOTP.1
```

## cardReset

Syntax:
**`cardReset`**

Reboots the VGMC. This command does not apply to the Signaling Server. You can execute this command from the VGMC and VxWorks shells. This is the preferred way to reboot the card because the application can reboot in a controlled manner (for example, the OM file is written and closed before the card reboots).

## cd

Syntax:
**`cd "dirPath"`**

Change the directory to the path specified by **`dirPath`**.

The A: drive is the faceplate PC card on the SMC and ITG-P and the floppy drive on the Signaling Serve. The C: drive is the internal flash drive on the ITG-P or SMC. For information about the directory structures of the various cards, see "VGMC" (page 20).

The following table describes the command parameters.

**Table 10**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| dirPath | "string" | Directory path to change to.<br><br>Drive letters must be capitalized. File names can be lowercase and must follow the 8.3 format. |

The following example shows command output.

```
-> cd "/C:"
value = 0 = 0x0
-> pwd
/C:
value = 4 = 0x4
->
```

## censusShow

Syntax:
**`censusShow`**

Display information about the cards in the node where you enter the command. For each card, the card platform type, TN, TLAN IP address and ELAN IP address appear.

Each card is scheduled to send a connection confirmation to peers once each minute. A TimeOut field indicates the number of times the current card missed a census message from a particular peer. If the timeout reaches 3, the card goes out of service, and the next censusShow does include it.

The censusShow output is in the electShow command output.

The following example shows command output.

```
-> censusShow
AutoAnnounce :  1
Timer duration :  60 (Next timeout in 3 sec)
====== all tps ======
Num Platform TN TLAN ELAN TimeOut
0 ISP 1100 0000 47.11.215.158 47.11.216.242 0
value = 0 = 0x0

-> censusShow
AutoAnnounce :  1
Timer duration :  60 (Next timeout in 55 sec)
====== all tps ======
Num Platform TN TLAN ELAN TimeOut
0 ITG Pentium 0410 47.11.215.65 47.11.216.70
0 1 ITG SA 0008 47.11.215.170 47.11.217.5 1
value = 0 = 0x0
```

### chkdsk

Syntax:
**`chkdsk "dev", repairLevel, entryLenType`**

The chkdsk command identifies corruption on the VGMC C: drive. The command checks the disk and prints a report of the results. While chkdsk runs, other routines and tasks cannot access the partition.

The following table describes the command parameters.

**Table 11**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| dev | – | Specify the drive to check.<br><br>Enter "/C:" when you check the VGMC internal flash disk. |
| repairLevel | 0 | Specify the action to take, where<br>0 = do not write to disk, only report errors found (default value) |
| entryLenType | 0–2 | Specify the condition of the disk, where<br>0 = disk not fully used or average file is more than 16 clusters (recommended)<br>1= average file is about 4 clusters in size<br>2 = disk is full, small files about 1 cluster in size on the average |

The following example shows command output.

```
-> chkdsk "/C:",0,0
Copyright (c) 1993-1996 RST Software Industries Ltd.
Israel.  All rights reserved

ver:  2.6 FCS

Disk Check In Progress ...

total disk space (bytes) :  3,923,968
bytes in each allocation unit :  2,048
total allocation units on disk :  1,916
bad allocation units :  0
```

```
available bytes on disk :  2,037,760
available clusters on disk :  995
maximum available contiguous chain (bytes) :  1,935,360
available space fragmentation (%) :  6
clusters allocated :  921
Done Checking Disk.
value = 0 = 0x0
->
```

## clearLeader

Syntax:

**`clearLeader`**

Set up the card as a Follower by deleting the ELAN IP address data from the NVRAM, clearing the Leader flag, and setting a flag so the BOOTP request is sent on reboot instead of reading the IP parameters from the NVRAM. This command does not apply to the Signaling Server.

The following example shows command output for the ITG-P (also applicable to SMC).

```
VGMC> clearLeader
This card will be a follower on reboot.
value = 41 = 0x29 = ')'
VGMC> NVRIPShow
IP address :  255.255.255.255
Gateway :  0.0.0.0
Subnet Mask:  0.0.0.0
Set as Follower.
Using bootp to acquire IP address.
value = 37 = 0x25 = '%'
```

## clrUengineStat

Syntax:

**`clrUengineStat chNum`**

This command clears the uEngine statistics for the specified channel. This command applies only to the SMC card.

The following table describes the command parameters.

**Table 12**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | 0–31 | Channel number |

```
-> clrUengineStat 8
value = 0 = 0x0
```

## configFileGet

Syntax:

**`configFileGet "srvrIP","userID","password","path","fil
ename"`**

This command transfers the file from the specified FTP server, copies it to the VGMC or Signaling Server CONFIG directory, backs up the existing config.ini file, renames the new file to CONFIG.INI and parses the codec information.

The `path` and `filename` parameters specify the path and file on the FTP server. For examples of user IDs and passwords for various FTP servers, see "swDownload " (page 258).

The following example shows command output.

```
VGMC> configFileGet "192.168.1.10","uid","pswd","c:/u/itgl","config.ini"
value = 0 = 0x0
VGMC>
FEB 14 15:04:26 tMAM: Info notifyConfigReadDone for 24 channels
FEB 14 15:04:26 tMAM: Info Initiate download request for codec 1
FEB 14 15:04:26 tMVX_DIM: Info Download codec image completed
-DSP# 0, Image Id= 0 ....
```

## configFilePut

Syntax:

**`configFilePut "hostIP","userID","password","path","fil
ename"`**

This command transfers the bootp.tab from the VGMC or Signaling Server CONFIG directory to specified host. The **`path`** and **`filename`** parameters specify the directory on the remote host and the name of the file to create. For examples of user IDs and passwords for various FTP servers, see "swDownload " (page 258).

The following example shows command output.

```
VGMC> configFilePut "192.168.1.15","itgadmin","itgadmin
","/C:/","CONFIG.1"
value = 0 = 0x0
VGMC>
FEB 15 12:53:19 tfx5:  Notice File transfer starting:
```

```
/C:/config/config.ini -> 19 2.168.1.15:/C://CONFIG.1
FEB 15 12:53:19 tfx5:  Notice File transfer completed:
/C:/config/config.ini -> 1 92.168.1.15:/C://CONFIG.1
```

## copy

Syntax:
**`copy "sourceFilename", "destFilename"`**

Copies a file from the sourceFilename to the destFilename.

If you do not enter the `destFilename` parameter, the file is copied to the standard output device (that is, the TTY display), providing an easy way to look at the contents of the text files (config.ini, bootp.tab, syslog.n, and omreport.nnn).

The following example shows command output.

```
-> copy "bootp.tab"
#10.1.1.10 #192.168.1.201
#version=ITGIPPHONE

.subnet1 :sm=255.255.255.0:gw=192.168.1.102:ts=10.1.1.10:hn:
1:tc=.subnet1:ha="00:60:38:01:19:4f":ip=192.168.1.201:lp=10.1.1.1
255.255.255.0 10.1.1.1:dn=003 0:to=1: value = 0 = 0x0 ->
```

## csvShow

Syntax:
**`csvShow`**

Indicates whether the Connect Server is enabled or disabled on the card.

The following example shows command output if the card is the node Master.

```
-> csvShow
Connection Service
==================
Src Port:  10.1.1.7:4100
Node:  1
Terminals:  0

Set csvBlkAddr usiBlkAddr State Type IPAddress Port
--- ---------- ---------- --------- ------- --------------
- ----
value = 0 = 0x0
```

The following example shows command output if the card is not the node Master.

```
-> csvShow
Connection Service
==================
Src Port: 10.1.1.7:4100
Node:
Terminals: 0

CSV is currently disabled
value = 26 = 0x1a
```

### d

Syntax:
**d [adr[,nunits[,width]]]**

Display memory.

The following table describes the command parameters.

**Table 13**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| adr | – | Starting address in normal hexadecimal format (0xnnnnnnnn), as a dereferenced variable name (&varName), or as a function name.<br><br>If omitted or zero, the next block of memory appears, starting where the last **d** command completed. |
| nunits | – | Specify the number of data words of size **width** to display.<br><br>Defaults to the previous value if omitted or zero. |
| width | 1, 2, 4, 8 | Default to the previous value if omitted or zero.<br><br>If invalid, 1 is used. |

The following example shows command output.

```
-> d 0xf0,16,2
000000f0:  2ada 0020 ef00 002f 2adf 0020 ef00 002f *.*
.../..* .../.*
00000100:  eb44 0020 ee00 03fa 1f88 0020 ee00 03fb *D.
.......  .....*
value = 21 = 0x15
```

```
-> d 0xf0,4
000000f0:  2ada 0020 ef00 002f *.* .../.........*
value = 21 = 0x15
-> d 0xf0, 4, 1
000000f0:  da 2a 20 00 *.* .............*
value = 21 = 0x15
-> d 0xf0,4,4
000000f0:  00202ada 002fef00 00202adf 002fef00 *.* .../..*
.../.*
value = 21 = 0x15
-> d 0xf0,4,8
000000f0:  002fef0000202ada 002fef0000202adf *.* .../..*
.../.*
00000100:  03faee000020eb44 03fbee0000201f88 *D. .......
.....*
value = 21 = 0x15
```

## devs

Syntax:
**devs**

Display all devices on the Signaling Server. The hard drive partitions are /p and /u. The floppy drive is device /f0. The CDROM drive is device /cd0.

The following example shows command output.

```
-> devs drv name
0 /null
1 /dev/rtc
2 /tyCo/0
2 /tyCo/1
3 /aioPipe
7 /bsp
9 nbvws042:
4 /p
10 /vio
11 /tgtsvr
4 /u
12 /cd0
4 /f0
13 /pty/pty00.S
14 /pty/pty00.M
13 /pty/pty01.S
14 /pty/pty01.M
13 /pty/pty02.S
14 /pty/pty02.M
13 /pty/pty03.S
```

```
14 /pty/pty03.M
3 /pipe/bootpd
3 /pipe/srv.6
3 /pipe/rudp
3 /pipe/srv.39
15 /locale
3 /pipe/srv.38
4 /ums
3 /pipe/srv.48
3 /pipe/srv.49
4 /webxml/
value = 25 = 0x19
```

This command gracefully disables the NRS DB.

### dumptab

Syntax:
**`dumptab`**

Display the contents of the BOOTP server database. The information this command displays matches the contents of the /u/config/bootp.tab file.

The following example shows command output.

```
-> dumptab
# main 2.4.3
# (null):  dump of bootp server database.
# Dump taken FRI NOV 29 11:06:26 2002
#
# Legend:  (see bootptab.5)
# first field -- hostname (not indented)
# bf -- bootfile
# bs -- bootfile size in 512-octet blocks
# cs -- cookie servers
# df -- dump file name
# dn -- domain name
# ds -- domain name servers
# ef -- extension file
# ex -- exec file (YORK_EX_OPTION)
# gw -- gateways
# ha -- hardware address
# hd -- home directory for bootfiles
# hn -- host name set for client
# ht -- hardware type
# im -- impress servers
# ip -- host IP address
# lg -- log servers
# lp -- LPR servers
# ms -- message size
# mw -- min wait (secs)
# ns -- IEN-116 name servers
# nt -- NTP servers (RFC 1129)
# ra -- reply address override
# rl -- resource location protocol servers
# rp -- root path
# sa -- boot server address
# sm -- subnet mask
# sw -- swap server
# tc -- template host (points to similar host entry)
# td -- TFTP directory
# to -- time offset (seconds)
# ts -- time servers
# vm -- vendor magic number
# yd -- YP (NIS) domain
# ys -- YP (NIS) servers
# Tn -- generic option tag n

1:\
:dn=4 0 4 0:\
:gw=47.11.254.1:\
:hn:\
:ht=1:ha="00:02:B3:86:2A:A6":\
:ip=47.11.255.13:\
:lp=192.168.2.3, 255.255.0.0, 192.168.2.1:\
:sm=255.255.254.0:\
:to=111:\
:ts=192.168.2.2:  2:\
:dn=12 0 4 0:\
:gw=47.11.254.1:\
:hn:\
:ht=1:ha="00:60:38:8E:2A:F3":\
:ip=47.11.255.17:\
:lp=192.168.2.4, 255.255.0.0, 192.168.2.1:\
:sm=255.255.254.0:\
:to=111:\
:ts=192.168.2.2:  3:\
:dn=16 0 3 0:\
:gw=47.11.254.1:\
:hn:\
:ht=1:ha="00:60:38:BD:B3:01":\
:ip=47.11.255.42:\
:lp=192.168.2.5, 255.255.0.0, 192.168.2.1:\
:sm=255.255.254.0:\
:to=111:\
:ts=192.168.2.2:  .subnet1:\
:gw=47.11.254.1:\
:hn:\
:sm=255.255.254.0:\
:ts=192.168.2.2:  value = 29 = 0x1d
```

Show the card TPS state, current master, and a list of online TPSs.

```
oam> electShow
Node ID : 3333
Node Master :  Yes
Up Time :  2 days, 3 hours, 56 mins, 7 secs
TN : 000 00 00 00
Host Type :  ISP 1100
TLAN IP Addr :  47.11.239.235
ELAN IP Addr :  47.11.254.209
Election Duration :  15
Wait for Result time :  35
Master Broadcast period :  30
===== master tps =====
Host Type TN TLAN IP Addr
ISP 1100 000 00 00 00 47.11.239.235
Next timeout :  29 sec AutoAnnounce :  1
Timer duration :  60 (Next timeout in 44 sec)
====== all tps ======
Num TN Host Type ELAN MAC TLAN IP Addr
ELAN IP Addr Up Time NumOfSets TimeOut
001 000 00 00 00 ISP 1100 00:02:b3:ee:24:7d 47.11.239.235
47.11.254.209 002 03:56:07 0 0
002 008 00 01 00 SMC 00:20:d8:d0:9f:57 47.11.239.232
47.11.254.204 002 03:55:55 0 1 003 004 00 01 00 SMC
00:20:d8:d0:99:cf 47.11.239.231
47.11.254.202 002 03:55:57 0 1 004 008 01 01 00 SMC
00:20:d8:d0:64:30 47.11.239.233
47.11.254.206 002 03:55:33 0 1
====== All cards in node configuration are registered
======
```

Enable the local NRS DB, putting it in service.

```
pdt> enlNRS
pdt> 16/12/04 16:36:12 LOG0006 DB: Switch local primary to
active.
```

Force the NRS DB to disable, putting it out of service.

```
pdt> forcedisNRS
pdt> 16/12/04 16:34:14 LOG0006 DB: Switch local Primary NRS
to OOS by force disable.
```

## dim stats DSP, DSPchannel (ITG-P only)

Syntax:

**`dim stats DSP, DSPchannel`**

Print statistics from the DSP during a call. The data from this command is useful to assess a voice quality problem.

Print only valid data on the ITG-P card. It prints zeroes for the various counters on the SMC. This caused by the various ways the SMC card handles the RTP data through the microEngines. Information similar to that which dim statistics prints appear on the SMC card if you use the command vgwChStat with the variable DimDspStat = 1.

Typically, the data prints for the DSP channel handling a particular call. Use vgwShow to determine which channel is in use by the call and divide the channel number by 3: the whole number is the DSP number and the remainder is the DSP channel. For example, vgwShow indicates the call is on channel 13, which corresponds to DSP 4 channel 1.

The following table describes the command parameters.

**Table 14**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| DSP | 0–7 | Specify the DSP on the 24-port card from which to retrieve data. |
| DSPchannel | 0–2 | Specify a particular channel on the DSP.<br><br>If omitted, prints data for all channels on that DSP. |

The following example shows command output for the channels on DSP 0 on an ITG-P.

```
MXP>dim stats 0
DSP 0, Chan 0 Stats:
from_dsp = 227229
from_dsp_bad_chn = 0
from_dsp_bad_len = 0
from_dsp_err = 0
to_net = 227229
to_net_err = 0
to_net_no_buff = 0
from_net = 227314
```

```
from_net_err = 0
from_net_bad_len = 0
from_net_bad_state = 0
to_dsp = 227314
to_dsp_bad_length = 0
to_dsp_no_tcid = 0
to_dsp_soguard_drop= 0
to_dsp_bad_state = 0
to_dsp_not_ready = 0
to_dsp_err = 0
to_dsp_queued = 108
to_dsp_de_queued = 108
to_dsp_queue_flush = 0
to_dsp_queue_drop = 0
to_dsp_queue_error = 0
dsp_not_responding = 0

DSP 0, Chan 1 Stats:
from_dsp = 0
from_dsp_bad_chn = 0
...  (same parameters as Chan 0)
dsp_not_responding = 0

DSP 0, Chan 2 Stats:
from_dsp = 131210
from_dsp_bad_chn = 0
...  (same parameters as Chan 0)
to_dsp_queue_error = 0
dsp_not_responding = 0 MXP>
```

## dim ver

Syntax:

**dim ver**

Displays the version of DSP firmware, the DSP type, the supported codecs, and features loaded on each VGMC DSP.

The following example shows command output.

```
MXP>dim ver
DSP 0:  version = 7.0.2.3, Voice & Fax, VPBX, C548F/C549F,
Codecs 0xffd2, Features 0xEB
DSP 1:  version = 7.0.2.3, Voice & Fax, VPBX, C548F/C549F,
Codecs 0xffd2, Features 0xEB
DSP 2:  version = 7.0.2.3, Voice & Fax, VPBX, C548F/C549F,
Codecs 0xffd2, Features 0xEB
DSP 3:  version = 7.0.2.3, Voice & Fax, VPBX, C548F/C549F,
Codecs 0xffd2, Features 0xEB
DSP 4:  version = 7.0.2.3, Voice & Fax, VPBX, C548F/C549F,
Codecs 0xffd2, Features 0xEB
DSP 5:  version = 7.0.2.3, Voice & Fax, VPBX, C548F/C549F,
Codecs 0xffd2, Features 0xEB
DSP 6:  version = 7.0.2.3, Voice & Fax, VPBX, C548F/C549F,
Codecs 0xffd2, Features 0xEB
DSP 7:  version = 7.0.2.3, Voice & Fax, VPBX, C548F/C549F,
Codecs 0xffd2, Features 0xEB
MXP>
```

Codecs is a bit field encoded as follows:

PCM_MU 0x8000

PCM_A 0x4000

ADPCM16 0x2000

ADPCM24 0x1000

ADPCM32 0x0800

ADPCM40 0x0400

EADPCM16 0x0200

EADPCM24 0x0100

EADPCM32 0x0080

EADPCM40 0x0040

G729 0x0020

G729AB 0x0010

G723_5_3 0x0008

G723_6_3 0x0004

FAX 0x0002

For the preceding example, the string indicates the DSP supports. Codecs include G.711 A/mu law, ADPCM and EADPCM @ 16,24,32,40 Kbps, G.729AB and FAX. Features include 128 milliseconds ECAN tail, Conference, FAX, T-38 FAX, Caller ID, RTP, and FRF11.

## dim cfg

Syntax:

**`dim cfg`**

Displays the configuration for each VGMC DSP.

The following example shows command output.

```
MXP>dim cfg
DIM config info:
DIM: Num dsps 8, Max chan per dsp 3, Total tcids 24
DIM: Voice Polled mode, To-tele queue depth 100
DIM 0: Dsp Type C549. Image Id loaded 0x0
DIM 0: Num channels 3, Num fsx_fsr 4
DIM 0: Clk multVGMCier 4, HPI Vox mapping Fifo
DIM 0: Power mgmt timer period 5000, Wake up interrupt mask 0
DIM 0: HW Companding A-Law, Serial Port Config 8 Bit
DIM 0: Sync Interrupt Enabled, Clock Out Enabled
DIM 0: HINT Disabled
DIM 0: BDX Delay Control: NO
DIM 0: HW Gain Control: NO ... repeated for each DSP on card
DIM 7: Dsp Type C549. Image Id loaded 0x0
DIM 7: Num channels 3, Num fsx_fsr 4
DIM 7: Clk multVGMCier 4, HPI Vox mapping Fifo
DIM 7: Power mgmt timer period 5000, Wake up interrupt mask 0
DIM 7: HW Companding A-Law, Serial Port Config 8 Bit
DIM 7: Sync Interrupt Enabled, Clock Out Enabled
DIM 7: HINT Disabled
DIM 7: BDX Delay Control: NO
DIM 7: HW Gain Control: NO
MXP>
```

## DimDspStat = 1/0

This is a variable. To print the DSP statistics printed by the tsm_stat_req_vp_delay, tsm_stat_req_error and tsm_stat_req_rx_tx commands at the end of any call using a gateway channel, enter DimDspStat = 1. If you enter these commands asynchronously at the VxWorks shell, enter DimDspStat = 1; if you enter DimDspStat = 0, these commands print no output.

When DimDspStat is assigned a value of 1, the DSP statistics print at the end of every gateway channel call. Consider how busy the card is before enabling this, as it prints a block of data every time a call completes until the flag is reset to 0. Be sure to reset the flag to 0 when you finish using the commands that printed DSP data.

This "end of call" mechanism provides a useful way to see if DSP errors occur during calls. No identifying information about the call (other than the TCID and timestamp) are given, so if you need information about a particular call, use vgwShow while the call is active to identify the channel.

Clear this by setting the DimDspStat variable to 0. The default is off; the state is not saved and returns to off if the card is reset or reboots.

The following example shows command output.

```
-> DimDspStat = 1
_DimDspStat = 0x346298:  value = 1 = 0x1
->
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, avg_playout_delay =
59
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, lost_packet_count = 0
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, replay_packet_count
= 0
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, idle_packet_count =
14
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, dropped_packet_count
= 0
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, rx_packet_count = 156
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, invalid_header_count
= 0
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, to_micro_overflow_co
unt = 0
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, lost_enh_packet_coun
t = 0
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, no_core_packet_count
= 0
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, rx_packet_count = 78
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, tx_packet_count = 81
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, silence_packet_count
= 0
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, min_jitter = 20
SEP 06 11:39:08 tMVX_DIM: Info TCID 0, max_jitter = 20
->
```

### DimECStat = 1/0

This is a variable. To print the DSP statistics normally printed by the tsm_stat_req_ecdbg command at the end of any call using a gateway channel, enter DimDspStat = 1. This provides a useful way to collect information about the echo canceller performance during a call. No identifying information about the call is given other than the TCID and timestamp, so if information about a particular call is needed, use the vgwShow command while the call is active to identify the channel.

Clear this by setting the DimECStat variable to 0. The default is off; the state is not saved and returns to off if the card is reset or reboots.

When DimECStat is assigned a value of 1, the ECAN statistics prints at the end of every gateway channel call. Consider how busy the card is before you enable this variable, as it prints a block of data every time a call completes until the flag is reset to 0. Be sure to reset the variable to 0 when you are finished using it.

An example of the information output when this command is entered follows.

```
-> DimECStat=1
_DimECStat = 0x34f27c: value = 1 = 0x1
->
SEP 27 08:28:41 tMVX_DIM: Info TSG: 0 Echo Canceller Debug Stats
SEP 27 08:28:41 tMVX_DIM: Info CNV state = 1
SEP 27 08:28:41 tMVX_DIM: Info Px level = 697226
SEP 27 08:28:41 tMVX_DIM: Info Py level = 0
SEP 27 08:28:41 tMVX_DIM: Info Pe level = 340
SEP 27 08:28:41 tMVX_DIM: Info Post update cnt = 836
SEP 27 08:28:41 tMVX_DIM: Info Attempt update cnt = 836
SEP 27 08:28:41 tMVX_DIM: Info Divergence cnt = 0
SEP 27 08:28:41 tMVX_DIM: Info ERLE bypass cnt = 328
SEP 27 08:28:41 tMVX_DIM: Info XTONE count = 6
SEP 27 08:28:41 tMVX_DIM: Info Fore switch cnt = 12
SEP 27 08:28:41 tMVX_DIM: Info Back switch cnt = 10
SEP 27 08:28:41 tMVX_DIM: Info Xidle count = 2825
SEP 27 08:28:41 tMVX_DIM: Info Other bypass cnt = 48
```

### dimPrintChannelInfo

Syntax:

**`dimPrintChannelInfo chNum`**

This query displays the data configured for the specified channel. Ensure that the data printed matches the data configured in the OTM/EM application for this card.

The following table describes the command parameters.

**Table 15**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number |

The following example shows command output.

```
-> dimPrintChannelInfo 3
rx_coding type:  68
tx_coding type:  68
rx vif size:  320
tx vif size:  320
encapsulation:  3
companding:  1
vad enable:  TRUE
vad threshold:  -17
vox nom delay:  80
vox max delay:  160
fax rate:  144
fax nom delay:  100
fax pkt size:  30
fax tx level:  -13
fax cd thresh:  2
fax_encap:  7
fax nat:  20
idle noise:  -6500
in gain:  -8
out gain:  0
tx_in_gain:  4
ec tail delay:  128
rtp.SSRC: AABBCCDD
rx rtp.payload:  18
tx rtp.payload:  18
```

The coding type indicates the codec used; it has the following values:

8 - G.711 A-law
9 - G.711 mu-law
68 - G.729A, G.729AB (the vad enable line item shows which is active)
80 - G.723 5.3K
81 - G.723 6.3K 129 - T38 (fax)
144 - G.711CC (G.711 clear channel - used for fax interworking with BCM)

The vif size indicates the number of bits to pack for each packet. This relates to the packet time and the codec type. The equations are:
G.711: vif size = packet time (milliseconds) * 64
G.729: vif size = packet time (milliseconds) * 8
G.723: 192 bits/pkt for 30 milliseconds payload

The DSP gain settings are set by the following items:
in gain - Tx (PCM > IP) gain register between ECAN and codec
out gain - Rx (IP > PCM) gain register
tx_in_gain - Tx (PCM > IP) gain register between PCM and ECAN

### dimPrintECC

Syntax:

**`dimPrintECC chNum, Filter`**

Prints a DSP channel echo canceller filter coefficients for the foreground or background filters. The filter coefficients are set by the ECAN algorithm to cancel the echo received from the TDM network. Each milliseconds of the ECAN has 8 coefficients (taps), so a 32 milliseconds ECAN tail, for example, has a total of 256 coefficients that you must print to see the full state of the ECAN. The printed data is usually only meaningful to an ECAN DSP engineer; it is typically collected for them. The coefficients are not cleared when the Open Loop detection removes the ECAN from the path; the last values present remain.

This command freezes the ECAN coefficients, requests the correct number of coefficients based on the ECAN tail length (from 256 to 1024 coefficients) and then unfreezes the ECAN. Because a 128 milliseconds ECAN tail length prints 1024 coefficients, 8 coefficients print for each line.

The following table describes the command parameters.

**Table 16**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | 0–23 or 0–31 | Channel number of the call on the VGMC |
| filter | – | 0 = foreground filter (default)<br>1 = background filter (if present) |

The following example shows command output.

```
-> dimPrintECC 0

TCID 0:  Echo Canceller coefficients (data length = 56)
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
.
.
.
[eight coefficients printed per msec of ECAN tail length]
.
.
.
```

```
TCID 0:  Echo Canceller coefficients (data length = 56)
16 -1 4 0 12 3 -5 -5
-13 -3 -25 7 2 0 25 3
9 -5 7 0 -1 -6 -8 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
TCID 0:  Echo Canceller coefficients (data length = 16)
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0
value = 0 = 0x0
```

## dnldFailShow

Syntax:

**`dnldFailShow numOfLine`**

Show the download failed status logged in the active or inactive UFTP log file.

The following table describes the command parameters.

**Table 17**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| numOfLine | – | Optional. Specify the number of lines to print.<br><br>When no argument is passed, the command shows the contents of all failed UFTP download events in the active/inactive UFTP log files. |

The following example shows command output.

```
oam> dnldFailShow
Active F/W download file:  /u/log/UFTPLOG0.TXT
----------------------------------------
12/29/03 19:58:41 F/W dnld fail:  (47.11.217.11) I2004 (F/W
not exist)
12/29/03 20:24:30 F/W dnld fail:  (47.11.217.12) I2004 (F/W
size is 0)
12/29/03 21:42:11 F/W dnld fail:  (47.11.217.15) I2002
(RUDP connection down)
12/29/03 22:17:40 F/W dnld fail:  (47.11.217.20) I2004
(Response time out)
inactive F/W download file:  /u/log/UFTPLOG1.TXT
----------------------------------------
12/28/03 19:58:41 F/W dnld fail:  (47.11.217.11) I2004
```

```
(RUDP connection down)
12/28/03 20:24:30 F/W dnld fail:  (47.11.217.12) I2004
(RUDP connection down)
12/28/03 21:42:11 F/W dnld fail:  (47.11.217.15) I2002
(RUDP connection down)
12/28/03 22:17:40 F/W dnld fail:  (47.11.217.20) I2004
(Response time out)
```

### dosFsConfigShow

Syntax:

**`dosFsConfigShow`**

Display information about the DOS file system on the C: drive. Use the information to determine the remaining free space on the disk.

The following example shows command output.

```
-> dosFsConfigShow

device name:  /p
total number of sectors:  4192902
bytes per sector:  512
media byte:  0xf8
# of sectors per cluster:  64
# of reserved sectors:  1
# of FAT tables:  2
# of sectors per FAT: 256 max
# of root dir entries:  512
# of hidden sectors:  63
removable medium:
false disk change w/out warning:  not enabled
auto-sync mode:  not enabled
long file names:  not enabled
exportable file system:  not enabled
lowercase-only filenames:  not enabled
volume mode:  O_RDWR (read/write) available space:
2105966592 bytes
max avail. contig space:  2094465024 bytes
value = 0 = 0x0
->
```

The following example shows command output on the Signaling Server.

```
-> dosFsConfigShow
device name:  /u
total number of sectors:  4194241
bytes per sector:  512
```

```
media byte:  0xf8
# of sectors per cluster:  64
# of reserved sectors:  1
# of FAT tables:  2
# of sectors per FAT: 256 max
# of root dir entries:  512
# of hidden sectors:  4209093
removable medium:  false
disk change w/out warning:  not enabled
auto-sync mode:  not enabled
long file names:  not enabled
exportable file system:  not enabled
lowercase-only filenames:  not enabled
volume mode:  O_RDWR (read/write)
available space:  2066087936 bytes
max avail.  contig space:  2057109504 bytes
value = 0 = 0x0
```

## dsetKMRQShow

Syntax:

**`dsetKMRQShow`**

Print the current state of the keymap download to the registered IP Phones.

The following table describes the data parameters printed by this command.

**Table 18**
**Data output**

| Parameter | Description |
|---|---|
| RTTavg | Indicates the average measured time for all telephones that received keymaps. |
| RTTexp | Expected measured time. The default value is 300 milliseconds.<br><br>This can be reduced, for instance, to achieve better performance on a particular system. For more information, see "Keymap download control" (page 419).<br><br>When **`RTTavg`** is less than **`RTTexp`**, download performance is better than expected. |

The following example shows command output.

```
-> dsetKMRQShow
There are totally 0 requests pending for keymap download
process
RTTavg = 112ms
RTTexp = 300ms
value = 30 = 0x1e
```

### dsetCadenceTableShow

Syntax:
**`dsetCadenceTableShow startingEntry, endingEntry`**

This command displays the Cadence table entries from the Cadence table currently on the card, which is downloaded from the Call Server. The parameters startingEntry and endingEntry specify the range of cadence entries to display, where startingEntry is in the range 0 to 255 and endingEntry is in the range startingEntry to 255.

Use this command to confirm the table was successfully downloaded from the CS 1000 to the card.

The following table describes the data parameters printed by this command.

**Table 19**
**Data output**

| Parameter | Description |
|---|---|
| Num | Cadence Table number |
| End | Tone end, where<br>0 = Off<br>1 = On<br>2 = Repeat |
| WTON | Tone associated with Cadence. WTON= YES, WTON= NO |
| C*n* | Cycle *n*, where *n* is 1,2,3,4,5. |
| E*n*On | Element *n* ON time, where n is 1,2,3,4,5. |
| E*n*Off | Element *n* OFF time, where n is 1,2,3,4,5. |
| Tone*n* | Element *n* Tone ID, where n is 1,2,3,4,5. |

The following example shows the default entries from a typical North American Option 11C system.

```
-> dsetCadenceTableShow 0,40
Num End WTON C1 C2 C3 C4 C5 E1On E1Off E2On E2Off E3On E3Off
E4On E4Off E5On
```

```
E5Off Tone1 Tone2 Tone3 Tone4 Tone5
0 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0
0 1 2 0 1 0 0 0 0 0x19a 0x320 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0
0 0 0 2 2 0 1 1 0 0 0 0x134 0x4c 0x134 0x4c 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 3 0 0 0 0 0 0 0 0xcd 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 4 2 0 1 1 0 0 0 0x66 0x66 0xcd 0x333 0x0 0x0 0x0
0x0 0x0 0x0 0 0 0 0 0 5 2 0 1 0 0 0 0 0x64 0x64 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0 0 0 0 0 6 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0 0 0 0 0 7 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0 0 0 0 0 8 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0 0 0 0 0 9 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0 0 0 0 0 10 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0 0 0 0 0 11 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0 0 0 0 0 12 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0 0 0 0 0 13 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0 0 0 0 0 14 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0 0 0 0 0 15 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0 0 0 0 0 16 2 0 1 0 0 0 0 0x64 0x64 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0x0 0 0 0 0 0 17 2 0 1 0 0 0 0 0x32 0x32 0x0 0x0
0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0 0 18 2 0 1 0 0 0 0 0xa 0xa 0x0 0x0
0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0 0 19 2 0 1 0 0 0 0 0x28 0x3c 0x0
0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0 0 20 0 0 0 0 0 0 0 0xf 0x0 0x0
0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0 0 21 0 0 0 0 0 0 0 0x14 0x0
0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0 0 22 0 0 0 0 0 0 0 0x14
0x14 0x14 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0 0 23 0 0 0 0 0 0 0
0x3c 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0 0 24 0 1 0 0 0 0
0 0x14 0x0 0x14 0x0 0x14 0x0 0x0 0x0 0x0 0x0 6 3 6 0 0 25 0 0 0 0
0 0 0 0xc8 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0 0 26 0 0 0
0 0 0 0 0x32 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0 0 27 0 0
0 0 0 0 0 0x190 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0 0 28
0 0 0 0 0 0 0 0x7d 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0 0 0 0
29 2 0 1 0 0 0 0 0x14a 0x46 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0 0 0
0 0 30 2 0 1 0 0 0 0 0x64 0x32 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0
0 0 0 0 31 2 0 1 0 0 0 0 0x19a 0x320 0x0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 32 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 33 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 34 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 35 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 36 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 37 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 38 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 39 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 40 0 0 0 0 0 0 0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0
0x0 0 0 0 0 0 value = 142 = 0x8e
```

## dsetShow

Syntax:

**`dsetShow [debugLevel]`**

Shows every IP Phone registered with the TPS, with information different from the isetShow command.

The parameter **`debugLevel`** controls which information prints. If you do not enter a value for this parameter, the command prints only the telephone TN (in hexadecimal format), public signaling IP address, private signaling IP address, hardware ID, and terminal type. This can be useful when you need to look up the TN of a telephone for the ssdTrace 0 command. However, if **`debugLevel`** = 4 or more, additional information prints that you can use to debug the problem.

The MAC address of the telephone is separated from the other Hardware ID information by dashes. The private signaling IP address prints when the telephone is detected as behind a NAT device.

The following example shows command output.

```
-> dsetShow
TN IP Address Private IP Addr Hardware ID TermType
---- --------------- --------------- --------------------
----------
6044 47.11.215.183 18-000ae402e283-6602 i2001
6004 47.11.179.168 18-006038b689e9-6600 i2004
6005 47.11.179.167 192.168.0.233 18-0060387602b9-6600
i2004

-> dsetShow 4
TN IP Address Private IP Addr Hardware ID TermType Emulator
Terminal SessCall SessApp
---- --------------- --------------- --------------------
---------- ---------- ---------- ---------- ----------
6004 47.11.217.239 18-006038b689e9-6600 i2004 0x0229b5b4
0x0229d4e0 0x0229d1b0 0x0229af74
6044 47.11.215.183 18-000ae402e283-6602 i2001 0x0227de94
0x02288888 0x0227f284 0x0227f218
6005 47.11.217.241 192.168.1.112 18-0060387602b9-6600
i2004 0x0228d008 0x02294d9c 0x02294e30 0x0229dc54
value = 0 = 0x0
```

## dsetToneTableShow

Syntax:

**`dsetToneTableShow startingEntry, endingEntry`**

Display the entries from the Tone table on the card, which is downloaded from the Call Server.

The following table describes the command parameters.

**Table 20**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| startingEntry | 0–255 | Specify the range of tone entries to display. |
| endingEntry | **`startingEntry`**–255 | Specify the range of tone entries to display. |

Use this command to confirm the table was successfully downloaded from the CS 1000 to the card.

The following table describes the data parameters printed by this command.

**Table 21**
**Data output**

| Parameter | Description |
|---|---|
| Num | Tone Table Number |
| Freq *n* | Tone *n* frequency, where *n* is 1,2,3,4,5. |
| dbn | Tone *n* volume, where *n* is 1,2,3,4,5. |

The following example shows the default entries (that is, not modified through LD 56) from a typical North American CS 1000 Option 11C system.

```
-> dsetToneTableShow 0,111
Num Freq 1 db1 Freq 2 db2 Freq 3 db3 Freq 4 db4 || Num Freq 1
db1 Freq 2 db2 Freq 3 db3 Freq 4 db4
0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 1 0x15e 0x17 0x1b8 0x17
0x0 0x0 0x0 0x0
2 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 3 0x1b8 0x17 0x0 0x0 0x0
0x0 0x0 0x0
4 0x15e 0x13 0x1b8 0x13 0x0 0x0 0x0 0x0 || 5 0x1b8 0x19 0x1e0
0x19 0x0 0x0 0x0 0x0
6 0x1e0 0x17 0x0 0x0 0x0 0x0 0x0 0x0 || 7 0x1e0 0x1e 0x26c
0x1e 0x0 0x0 0x0 0x0
8 0x3fc 0x10 0x0 0x0 0x0 0x0 0x0 0x0 || 9 0x258 0x17 0x0 0x0
0x0 0x0 0x0 0x0
10 0x258 0x10 0x0 0x0 0x0 0x0 0x0 0x0 || 11 0x1b8 0x16 0x1e0
0x16 0x0 0x0 0x0 0x0
12 0x15e 0x17 0x1e0 0x17 0x0 0x0 0x0 0x0 || 13 0x1b8 0x18
0x26c 0x18 0x0 0x0 0x0 0x0
```

```
14 0x3ac 0xc 0x65e 0xa 0x0 0x0 0x0 0x0 || 15 0x2bc 0xc 0x4ba
0xa 0x0 0x0 0x0 0x0
16 0x2bc 0xc 0x53c 0xa 0x0 0x0 0x0 0x0 || 17 0x2bc 0xc 0x5c8
0xa 0x0 0x0 0x0 0x0
18 0x302 0xc 0x4ba 0xa 0x0 0x0 0x0 0x0 || 19 0x302 0xc 0x53c
0xa 0x0 0x0 0x0 0x0
20 0x302 0xc 0x5c8 0xa 0x0 0x0 0x0 0x0 || 21 0x352 0xc 0x4ba
0xa 0x0 0x0 0x0 0x0
22 0x352 0xc 0x53c 0xa 0x0 0x0 0x0 0x0 || 23 0x352 0xc 0x5c8
0xa 0x0 0x0 0x0 0x0
24 0x3ac 0xc 0x53c 0xa 0x0 0x0 0x0 0x0 || 25 0x3ac 0xc 0x4ba
0xa 0x0 0x0 0x0 0x0
26 0x3ac 0xc 0x5c8 0xa 0x0 0x0 0x0 0x0 || 27 0x2bc 0xc 0x65e
0xa 0x0 0x0 0x0 0x0
28 0x302 0xc 0x65e 0xa 0x0 0x0 0x0 0x0 || 29 0x352 0xc 0x65e
0xa 0x0 0x0 0x0 0x0
30 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 31 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0
32 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 33 0x190 0x13 0x0 0x0 0x0
0x0 0x0 0x0
34 0x17c 0x1b 0x190 0x13 0x2a 0x1b 0x0 0x0 || 35 0x0 0x0 0x0
0x0 0x0 0x0 0x0 0x0
36 0x2bc 0x11 0x4ba 0xf 0x0 0x0 0x0 0x0 || 37 0x2bc 0x11 0x53c
0xf 0x0 0x0 0x0 0x0
38 0x2bc 0x11 0x5c8 0xf 0x0 0x0 0x0 0x0 || 39 0x302 0x11 0x4ba
0xf 0x0 0x0 0x0 0x0
40 0x302 0x11 0x53c 0xf 0x0 0x0 0x0 0x0 || 41 0x302 0x11 0x5c8
0xf 0x0 0x0 0x0 0x0
42 0x352 0x11 0x4ba 0xf 0x0 0x0 0x0 0x0 || 43 0x352 0x11 0x53c
0xf 0x0 0x0 0x0 0x0
44 0x352 0x11 0x5c8 0xf 0x0 0x0 0x0 0x0 || 45 0x3ac 0x11 0x53c
0xf 0x0 0x0 0x0 0x0
46 0x3ac 0x11 0x4ba 0xf 0x0 0x0 0x0 0x0 || 47 0x3ac 0x11 0x5c8
0xf 0x0 0x0 0x0 0x0
48 0x2bc 0x11 0x65e 0xf 0x0 0x0 0x0 0x0 || 49 0x302 0x11 0x65e
0xf 0x0 0x0 0x0 0x0
50 0x352 0x11 0x65e 0xf 0x0 0x0 0x0 0x0 || 51 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0
52 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 53 0x514 0xd 0x5dc 0xd
0x0 0x0 0x0 0x0
54 0x2bc 0xd 0x384 0xd 0x0 0x0 0x0 0x0 || 55 0x2bc 0xd 0x44c
0xd 0x0 0x0 0x0 0x0
56 0x384 0xd 0x44c 0xd 0x0 0x0 0x0 0x0 || 57 0x2bc 0xd 0x514
0xd 0x0 0x0 0x0 0x0
58 0x384 0xd 0x514 0xd 0x0 0x0 0x0 0x0 || 59 0x44c 0xd 0x514
0xd 0x0 0x0 0x0 0x0
60 0x2bc 0xd 0x5dc 0xd 0x0 0x0 0x0 0x0 || 61 0x384 0xd 0x5dc
```

```
0xd 0x0 0x0 0x0 0x0
62 0x44c 0xd 0x5dc 0xd 0x0 0x0 0x0 0x0 || 63 0x2bc 0xd 0x6a4
0xd 0x0 0x0 0x0 0x0
64 0x384 0xd 0x6a4 0xd 0x0 0x0 0x0 0x0 || 65 0x44c 0xd 0x6a4
0xd 0x0 0x0 0x0 0x0
66 0x514 0xd 0x6a4 0xd 0x0 0x0 0x0 0x0 || 67 0x5dc 0xd 0x6a4
0xd 0x0 0x0 0x0 0x0
68 0x190 0xb 0x0 0x0 0x0 0x0 0x0 0x0 || 69 0x190 0xe 0x0 0x0
0x0 0x0 0x0 0x0
70 0x190 0xe 0x0 0x0 0x0 0x0 0x64 0x0 || 71 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0
72 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 73 0x15e 0xf 0x1b8 0xf
0x0 0x0 0x0 0x0
74 0x1e0 0xf 0x26c 0xf 0x0 0x0 0x0 0x0 || 75 0x1b8 0xf 0x1e0
0xf 0x0 0x0 0x0 0x0
76 0x190 0x19 0x0 0x0 0x0 0x0 0x0 0x0 || 77 0x190 0xe 0x1c2
0xe 0x0 0x0 0x0 0x0
78 0x1e0 0x13 0x26c 0x13 0x0 0x0 0x0 0x0 || 79 0x1b8 0x13
0x1e0 0x13 0x0 0x0 0x0 0x0
80 0x1e0 0x13 0x0 0x0 0x0 0x0 0x0 0x0 || 81 0x1a4 0x9 0x0 0x0
0x0 0x0 0x0 0x0
82 0x1b8 0x1d 0x0 0x0 0x0 0x0 0x0 0x0 || 83 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0
84 0x15e 0x11 0x1b8 0x11 0x0 0x0 0x0 0x0 || 85 0x190 0x11
0x1c2 0x11 0x0 0x0 0x0 0x0
86 0x190 0x11 0x0 0x0 0x0 0x0 0x0 0x0 || 87 0x578 0x1a 0x0 0x0
0x0 0x0 0x0 0x0
88 0x3b6 0xc 0x0 0x0 0x0 0x0 0x0 0x0 || 89 0x578 0xc 0x0 0x0
0x0 0x0 0x0 0x0
90 0x708 0xc 0x0 0x0 0x0 0x0 0x0 0x0 || 91 0x1d6 0x0 0x0 0x0
0x0 0x0 0x0 0x0
92 0x3ac 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 93 0x514 0x0 0x0 0x0
0x0 0x0 0x0 0x0
94 0x5dc 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 95 0x758 0x0 0x0 0x0
0x0 0x0 0x0 0x0
96 0x15e 0xa 0x1b8 0xa 0x0 0x0 0x0 0x0 || 97 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0
98 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 99 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0
100 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 101 0x258 0x13 0x0 0x0
0x0 0x0 0x0 0x0
102 0x320 0x13 0x0 0x0 0x0 0x0 0x0 0x0 || 103 0x578 0x17 0x0
0x0 0x0 0x0 0x0 0x0
104 0x334 0x7 0x0 0x0 0x0 0x0 0x0 0x0 || 105 0x0 0x0 0x0 0x0
0x0 0x0 0x0 0x0
106 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 107 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0
```

```
108 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 109 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0
110 0x0 0x0 0x0 0x0 0x0 0x0 0x0 0x0 || 111 0x0 0x0 0x0 0x0 0x0
0x0 0x0 0x0
value = 60 = 0x3c = '<'
->
```

## DsetSideToneEnable = 1/0

By default, this variable is 1. To turn off sidetone (that is, hearing your own voice in the handset or headset) on all calls made by IP Phones registered with that TPS , enter DsetSideToneEnable = 0. The variable is reset to 1 upon card reboot.

The following example shows command output.

```
-> DsetSideToneEnable = 1
_DsetSideToneEnable = 0x37b558:  value = 1 = 0x1
```

## dsetVosboShow

Syntax:
**`dsetVosboShow emulatorAddress`**

Print the Virtual Office and Branch Office feature related DSET status information for a particular IP Phone.

The following table describes the command parameters.

**Table 22**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| emulatorAddress | – | Emulator address of the IP Phone.<br><br>You can retrieve this IP address by entering **`dsetShow 4`** at the vxWorksShell prompt. |

The following example shows command output.

```
-> dsetShow 4
TN IP Address Hardware ID TermType Emulator Terminal
SessCall SessApp
---- -------------- ----------------- ----------
---------- ---------- ---------- --------
6005 10.1.1.10 1800603876c79d6600 i2004 0x09e4a240
0x09e4c4a0 0x09e4af38 0x09e4d1a8
0000 10.1.1.13 180060387638e06600 i2004 0x09d26a08
0x09e49ee8 0x09d2742c 0x09e49058
6006 10.1.1.12 18000638dd06116600 i2002 0x09d21894
```

```
0x09d23a64 0x09d22534 0x09d246e0
value = 0 = 0x0

-> dsetVosboShow 0x09e4a240
sVOBUSupported=1
reg type=[Virtual]
VOSBO Status:  Login
User ID: 2041
Password:  1234
NPI / TON: 0 / 0
Home TPS IP: 10.1.1.6
Remote TPS IP: 0.0.0.0
Main TPS IP: 0.0.0.0
DSET VOSBO Status:  None
DSET Error Rate:  0
DSET voHostTermType:  2
DSET voHostHwid:  180060387638e06600
DSET BUID: MOTN
TN: 0x6004
MOTN TYPE: 0x2
UserDisplayFlag:  1
voUserInvalidIDReason:  0
i = 0:  Virtual Office Login --- pdset->hAppControl[i] =
0x9e4cbc8 the flag isNotPhoneOption is 0:
i = 1:  Virtual Office Logout --- pdset->hAppControl[i] =
0x9e4cb74 the flag isNotPhoneOption is 0:
i = 2:  Branch User Config --- pdset->hAppControl[i] =
0x9e4cb20 the flag isNotPhoneOption is 0:
i = 3:  Resume Normal Mode --- pdset->hAppControl[i] =
0x9e4cacc the flag isNotPhoneOption is 0:
i = 4:  Test Local Mode --- pdset->hAppControl[i] =
0x9e4ca78 the flag isNotPhoneOption is 0:
Text Editor --- pdset->hAppControl[5] = 0x0
Message Box --- pdset->hAppControl[6] = 0x0
value = 0 = 0x0
->
```

## DSPReset

Syntax:

**`DSPReset DSP`**

Reset the specified DSP. This command applies only to the VGMCs. All associated channels are closed. The codec image is then downloaded to the DSP and the channels are reregistered with the Call Server. An active call on any channel of the DSP is released.

The following example shows command output.

```
value = 0 = 0x0
VGMC> DSPReset 0
MAR 13 19:56:46 MAM: Info Reset DSP 0tMVX_SPY: Info
0152178642 - DIM: 0:*, DSP
BUSY -- No status response Message

tMVX_SPY: Info 0152178642 - DIM: 0:*, BRINGING DSP DOWN !!

MAR 13 19:56:48 tMVX_DIM: Error ITG2034 DSP channel
unexepectedly closed:  0 (44)
MAR 13 19:56:48 tMVX_DIM: Error ITG2034 DSP channel
unexepectedly closed:  1 (44)
MAR 13 19:56:48 tMVX_DIM: Error ITG2034 DSP channel
unexepectedly closed:  2 (44)
MAR 13 19:56:48 tMVX_DIM: Error ITG2025 DSP download:
failed - Retry 0 (202)
MAR 13 19:56:48 VGW: Error channel 0, unexpectedly closed
(reason -2)
MAR 13 19:56:48 VGW: Info VGW offline announce channel 0
MAR 13 19:56:48 tMVX_DIM: Info Download codec image
completed -DSP# 0, Image Id = 0
MAR 13 19:56:48 VGW: Error channel 1, unexpectedly closed
(reason -2)
MAR 13 19:56:48 VGW: Info VGW offline announce channel 1
MAR 13 19:56:49 VGW: Error channel 2, unexpectedly closed
(reason -2)
MAR 13 19:56:49 VGW: Info VGW offline announce channel 2 MAR
13 19:56:49 VGW: Info Registering channels 0 to 2
MAR 13 19:56:49 VGW: Info Channel 0, already registered with
CS (?)
MAR 13 19:56:49 VGW: Info Channel 1, already registered with
CS (?)
MAR 13 19:56:49 VGW: Info Channel 2, already registered with
CS (?)
```

### e2dsetShow emulatorAddress

Syntax:

**`e2dsetShow emulatorAddress`**

Print information regarding the states and current display of an IP Phone. It is useful if you have no physical access to the IP Phone but want to see the displayed information.

The first block of information displays Set Based Installation (SBI) related data. This command can also be useful when you work on an IP Phone installation problem.

When a call is active on the telephone, the callProcState becomes x11cpsActive, and the e2AudioStreamState becomes RxTxOpen. Also, the e2Lamp value for the DN with the active call changes from 0 to 1.

The following table describes the command parameters.

**Table 23**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| emulatorAddress | – | Emulator address of the IP Phone.<br><br>To retrieve this IP address, enter **`dsetShow 4`** at the vxWorksShell prompt. |

> ***Note:*** Improper use of this command (that is, passing an incorrect IP address) can cause an exception. Ensure that you enter the correct emulator address before you press Enter.

The following example shows command output.

```
-> dsetShow 4
TN IP Address Hardware ID TermType Emulator Terminal
Session
---- --------------- ------------------ --------
---------- ---------- ----------
6005 10.1.1.5 180060387641f36600 i2002 0x013db124
0x01a96cd8 0x01a961d8
6004 10.1.1.4 180060387641c16600 i2004 0x013d8f58
0x01a95fa8 0x0185d10c
6007 10.1.1.6 180060387606586600 i2050 0x013d4e5c
0x013d72cc 0x013d6634
value = 0 = 0x0
->
-> e2dsetShow 0x013d8f58
=== SBI Data ===
isSBIset 0
isSBIhandfreeKeyOn 0
isSBIheadSetKeyO 0
isSBIsetGoOffHook 0
=== states ===
isRegistered 1
e2State e2StateIdle, localState e2LocalStateNone,
callProcState x11cpsIdle, cpndState cpndIdle,
e2AudioStreamState RxTxClose
ActiveDNKey 255
=== display overlays ===
overlayState 0
line1 <>
line2 <>
line3 <>
=== display lines ===
displayState 0
line1 <>
line2 <>
line3 <>
current cursor:  [0, 4]
editBuffer <>
=== soft keys ===
key counts = 11, current page = 0
Text m1Key e2Key m1Lamp e2Lamp || Text m1Key e2Key m1Lamp
e2Lamp || Text m1Key e2Key m1Lamp e2Lamp || Text m1Key e2Key
m1Lamp e2Lamp ||
<Trans> 17 0 0 0 ||<Conf> 18 1 0 0 ||<Forward> 19 2 0 0
||<More...> 0 3 15 1 ||
<RingAgn> 20 4 0 0 ||< Park> 21 5 0 0 ||<Pickup> 22 6 0 0
||<More...>7 15 1 ||
<PrivRls> 24 8 0 0 ||<Charge> 25 9 0 0 ||<CParty> 26 10 0 0
||<More...>0 11 15 1 ||
< > 0 12 15 1 || < > 0 13 15 1 || < > < > 0 14 15 1 | < > 0 15 15 1 ||

=== feature keys ===
current page 0
Text m1Key e2Key m1Lamp e2Lamp || Text m1Key e2Key m1Lamp
e2Lamp || Text m1Key e2Key m1Lamp e2Lamp || Text m1Key e2Key
m1Lamp e2Lamp || Text m1Key e2Key m1Lamp e2Lamp || Text
m1Key e2Key m1Lamp e2Lamp ||
< 1111> 0 35 15 1 || < > 1 36 0 0 || < > 2 37 0 0 || < > 3 38 0 0 ||
< > 4 39 0 0 || < > 5 40 0 0 || < > 2 35 0 0 ||< > 3 36 0 0 || < > 4
37 0 0 || < > 5 38 0 0 || < > 6 39 0 0 ||7 40 0 0 ||

=== local soft keys ===
0 = <> 1 = <> 2 = <> 3 = <>
value = 1 = 0x1
->
```

## echoServerShow

Syntax:

**`echoServerShow action`**

Print configuration information about the ESs and information from a particular LTPS on the phone interactions with the ESs. Use this command on an LTPS card to investigate a problem with a telephone registered to that LTPS card, or to uncover patterns of communication problems between phones and ESs.

The following table describes the command parameters.

**Table 24**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| action | 99 | Optional.<br><br>When you enter echoServerShow 99, the counter values are reset after they are displayed. When you enter only echoServerShow, the counter values are displayed without being reset. |

The following table describes the data parameters output by this command and how to interpret them.

**Table 25**
**Data output**

| Parameter | Description |
|---|---|
| Configured | The IP address and port configured for this ES in LD 117. |
| Actual | IP address and port for this ES, followed by an explanation in parenthesis. This differs from the Configured parameter if the default address (0.0.0.0) is configured.<br><br>The explanation in parenthesis is one of the following:<br><br>• (TLAN IP, this card) = IP address used is the TLAN port of this card; the ES is active on this card.<br>• (node IP, this card) = IP address used is the node IP; ES is active on this card because it is the node master.<br>• (node IP, other card) = IP address used is the node IP, but some other card is currently the node master; ES is not active on this card.<br>• (not this card) = IP address is neither this card TLAN IP or the node IP address; ES is not active on this card. |
| LTPS request sent | Number of Resolve Port Mapping Request messages sent from the LTPS to IP Phones with this ES identified as the one to contact. |
| Failed resp rec'd | Number of Resolve Port Mapping Ack messages received from IP Phones having the public IP address and port set to 0.0.0.0:0000. Each increment of this counter indicates a telephone never received the Discover Port Mapping Ack response from the ES (that is, all 10 attempts failed). |

The following example shows command output.

```
->echoServerShow
Echo Server 1
------------------------------------------------
Configured:  0.0.0.0:10000
Actual:  47.11.212.54:10000 (TLAN IP, this card)
LTPS request sent:  112665
Failed resp rec'd:  0

Echo Server 2
------------------------------------------------
Configured:  0.0.0.0:10000
Actual:  47.11.212.60:10000 (node IP, other card)
LTPS request sent:  82201
Failed resp rec'd:  0
NAT Timeout:  30 seconds
```

When you enter the reset parameter, the counter values are displayed but then reset to 0 for the next time you enter the command.

```
->echoServerShow 99
Echo Server 1
------------------------------------------------
Configured:  0.0.0.0:10000
Actual:  47.11.212.54:10000 (TLAN IP, this card)
LTPS request sent:  81563
Failed resp rec'd:  40

Echo Server 2
------------------------------------------------
Configured:  0.0.0.0:10000
Actual:  47.11.212.60:10000 (node IP, other card)
LTPS request sent:  50199
Failed resp rec'd:  4
NAT Timeout:  30 seconds
Counters reset

->echoServerShow
Echo Server 1
------------------------------------------------
Configured:  0.0.0.0:10000
Actual:  47.11.212.54:10000 (TLAN IP, this card)
LTPS request sent:  0
Failed resp rec'd:  0

Echo Server 2
```

```
---------------------------------------------
Configured:  0.0.0.0:10000
Actual:  47.11.212.60:10000 (node IP, other card)
LTPS request sent:  0
Failed resp rec’d:  0
NAT Timeout:  30 seconds
```

## electShow

Syntax:

**`electShow`**

Print details about the node mastership election. This command displays the information from the censusShow command, plus additional information about the node Mastership election.

The output first displays information about the card the command was run on. This is followed by a list of cards in the node. For each card, the card platform type, TN, TLAN IP address, and ELAN IP address print.

Each card is scheduled to send a connection confirmation message to peers once each minute. A TimeOut field indicates the number of times the current card missed a census message from a particular peer. If the timeout reaches 3, the card is considered to be out of service, and the next censusShow does not include it.

The following example shows command output for the node Master card.

```
-> electShow

Node ID : 3918
Is master :  1
Up Time(sec) :  169634
TN : 0000 Platform :  ISP 1100
IP TLAN : 47.11.215.44
IP ELAN : 47.11.217.158
Election Duration :  2
Wait for Result time :  35
Master Broadcast period :  30
===== master tps =====
PlatForm TN TLAN
ISP 1100 0000 47.11.215.44
Next timeout = 20 sec
AutoAnnounce :  1
Timer duration :  60 (Next timeout in 26 sec)
====== all tps ======
Num Platform TN TLAN ELAN TimeOut
0 ISP 1100 0000 47.11.215.44 47.11.217.158 0
```

```
1 ITG SA 0408 47.11.215.30 47.11.216.246 0
2 ITG Pentium 0410 47.11.215.159 47.11.216.181 0
value = 0 = 0x0
```

The following example shows command output for a non-Master card in the node.

```
VGMC> electShow
Node ID : 3918
Is master :  0
Up Time(sec) :  167777
TN : 0408
Platform :  ITG SA
IP TLAN : 47.11.215.30
IP ELAN : 47.11.216.246
Election Duration :  2
Wait for Result time :  35
Master Broadcast period :  30
===== master tps =====
PlatForm TN TLAN
ISP 1100 0000 47.11.215.44
Next timeout = 8 sec
AutoAnnounce :  1
Timer duration :  60 (Next timeout in 41 sec)
====== all tps ======
Num Platform TN TLAN ELAN TimeOut
0 ITG SA 0408 47.11.215.30 47.11.216.246 0
1 ISP 1100 0000 47.11.215.44 47.11.217.158 0
2 ITG Pentium 0410 47.11.215.159 47.11.216.181 1
value = 0 = 0x0
```

## eStatShow (printing statistics)

Syntax:

**`eStatShow "source"`**

Request Ethernet statistics for an IP Phone for the VGMC. The IP Phone returns the response message to the VGMC. This command is supported by Phase 2 IP Phones and the IP Softphone 2050. Execute this command from the VGMC> prompt.

The following example shows command output.

```
Ethernet Statistic Report from Set (47.11.215.153)
100 base T full duplex
Auto negotiate protocol received
VLAN ID: 88
Priority:  1
```

```
Packet collisions:  100
CRC errors:  30
Framing Errors:1
```

### eStatShow (clearing statistics)

Syntax:
**`eStatShow "source", "clear"`**

Clear the Ethernet statistics count when count is 1.

The following example shows command output.

```
->eStatShow "47.11.213.216"
value = 247380368 = 0xebeb990
->eStatShow Report from set (47.11.213.216):
Duplex Mode:  0
Auto Negotiate Protocol Received:  0x3
Interface Speed:  1
VLAN Priority Bit:  0
VLAN ID: 1
Packet Collision Peg Count:  3
CRC Error Peg Count:  1
Frame Error Peg Count:  8
```

### exit

Syntax:
**`exit`**

Exit the current shell and return to the next higher shell. On the Signaling Server, you can use this command to go from the vxshell to the pdt> shell to the oam> shell.

The following example shows command output.

```
-> exit
pdt> exit

oam>
```

On the VGMC, you can use this command to go from the VxWorks shell to the VGMC> shell.

```
-> exit
VGMC>
```

### firmwareFileGetI2002

Syntax:

**`firmwareFileGetI2002 "srvrIP", "uid", "passwd", "path", "fname"`**Syntax:

Download the IP Phone 2002 firmware from the specified FTP server, uncompress the firmware, and upgrade the UMS policy.

For more information about the parameters used in this command, see “firmwareFileGetI2004 ” (page 128). This function internally calls firmwareFileGetI2004, so you can use that command directly to download either firmware file.

### firmwareFileGetI2004

Syntax:

**`firmwareFileGetI2004 "srvrIP", "userID", "password", "path", "filename"`**

Download the IP Phone 2004 firmware from the specified FTP server, uncompress the firmware, and upgrade the UMS policy. The command executes from the VxWorks CLI.

This command determines the firmware for the terminal type by parsing the input file name. If it is a Signaling Server, the firmware is downloaded from the remote host through FTP without compressing the file on the fly. If it is a VGMC, the command checks the disk space on the flash drives (C: and A:), and if enough space is available, the command continues to download the firmware from the remote FTP host and compresses the file on the fly.

The following table describes the command parameters.

**Table 26**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| srvrIP | – | IP address of the FTP server. |
| userID | – | Server logon user ID. |
| password | – | Server logon password. |
| path | – | Path on the server of the file to retrieve.<br>Drive letters must be capitalized, for example, C:/path. |
| filename | – | File name of the file to retrieve.<br>Must be in 8.3 format. |

The following example shows command output.

```
->firmwareFileGet
"47.11.229.34","joeuser","joeuser_password","/C:/Firmwa
reFilez/i2004/","CA02B26"
value = 0 = 0x0

APR 09 10:00:37 tUMS: Info File transfer completed
APR 09 10:00:37 tUMS: Info Set new FW file location from
/C:/FW/FWFILE.1 to /C:/FW/FWFILE.1
APR 09 10:00:37 tUMS: Info FW file uncompress
APR 09 10:00:37 tUMS: Info Get FW version CA02B26 from
uncompressed file /ums/fwfile
APR 09 10:00:37 tUMS: Info Resync IniFile from version
3002B02 to CA02B26
```

## firmwareFilePutI2002

Syntax:

**`firmwareFilePutI2002 "hostIP", "uid", "passwd", "path", "fname"`**

This command transfers the IP Phone 2002 firmware from the local directory to the specified FTP server directory and file name. For more information about the command parameters, see “firmwareFilePutI2004 ” (page 129).

## firmwareFilePutI2004

Syntax:

**`firmwareFilePutI2004 "hostIP", "uid", "passwd", "path", "fname"`**

This command transfers the IP Phone 2004 firmware from the local directory to the specfied host and creates a file in the specified directory. For examples of user IDs and passwords for various FTP servers, see “swDownload ” (page 258).

The following table describes the command parameters.

**Table 27**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| hostIP | – | Remote host to upload the firmware file to. |
| userID | – | Remote host logon user ID. |
| password | – | Remote host logon password. |
| path | – | Path to place the file on the remote host. |
| filename | – | file name. |

### firmwareVersionShow

Syntax:

Print the VGMC firmware version. Ensure that the firmware is Release 5.7 or later on the ITG-P and, Release 6.4 or later on the SMC.

The following example shows command output.

```
VGMC> firmwareVersionShow
Firmware Version = ITG Firmware Rls 4.0
value = 40 = 0x28 = '('
```

The command is not available on the Signaling Server and outputs the following:

```
-> firmwareVersionShow
No firmware version available.
value = 31 = 0x1f
```

### flashConfigShow

Syntax:

**`flashConfigShow`**

Displays information about the VGMC flash memory configuration. Execute this command from the BIOS or VxWorks shells. The command does not apply to the Signaling Server.

The following example shows command output.

```
-> flashConfigShow
Flash Vendor ID : 0x89
Flash device ID : 0x15
Flash device type :  i28f640j5
Flash base addr :  0xf9800000
Flash sector count :  64
Flash device width :  2
Flash chip count :  1
Flash device size :  8MB
value = 0 = 0x0
```

### ftpTypeA, ftpTypeI

Syntax:

**`ftpTypeA`**
**`ftpTypeI`**

These commands determine the type of FTP file transfer. The ftpTypeA command configures the transfer type as ASCII (the command default, appropriate for the config.ini, bootp.tab, omreport.nnn and syslog.n files), while ftpTypeI configures it as binary. Use the ftpTypeI command to configure binary mode if you transfer binary files from the VxWorks shell (for example, the VGMC application binary or IP Phone firmware).

### ftpXferSet

Syntax:

**`ftpXferSet`**

Configures the mode of file transfer for the hostFileGet command to FTP (the default is ITG mode).

### ftpVerbose = 0/1

This variable enables or disables the verbose mode of the FTP application. To print additional information to debug file transfer problems, enter ftpVerbose = 1. The default is 0 (off). This variable is cleared on reboot or when power is cycled to the card.

The following example shows command output.

```
-> ftpVerbose = 1
_ftpVerbose = 0x3903ac:  value = 1 = 0x1
-> hostFilePut 0,0,"192.168.1.14","itgadmin","itgadmin",
"/C:","CONFIG.1","CONFIG.INI"
220 VxWorks (5.3.1) FTP server ready 331 Password required
230 User logged in
200 Type set to I, binary mode

FEB 21 11:06:36 tShell:  Notice File transfer starting:
CONFIG.INI ->
192.168.1.14:/C:/CONFIG.1200 Po
rt set okay
150 Opening BINARY mode data connection
226 Transfer complete

FEB 21 11:06:37 tShell:  Notice File transfer completed:
CONFIG.INI ->
192.168.1.14:/C:/CONFIG.1221 B
ye...see you later
value = 0 = 0x0
```

### genToneOn chNum, side, freq, duration

Syntax:

**`genToneOn chNum, side, freq, duration`**

Enter this command on the VGMC to generate a tone from the DSP to the IP Phone or from the DSP to the TDM network. This command can be useful when you debug speech path problems as a known good point of reference to either end.

The following table describes the command parameters. Omitted parameters use a value of 0.

**Table 28**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| side | 0,1 | Direction in which tone is generated, where:<br>0 = TDM<br>1 = TLAN |
| freq | 0–3 | Tone frequency, calculated as frequency = 500*(freq+1)Hz . For example, default = 0 produces 500 Hz tone. |
| duration | 0–60 | Tone duration in seconds. 0 = indefinitely. |

The following example shows command output with all parameters entered, and command defaults with no parameters entered.

```
-> genToneOn 0,1,2,15
Tone generated to IP, channel:  0, freq:  1500, duration:
15
value = 0 = 0x0

-> genToneOn

Tone generated to TDM, channel:  0, freq:  500, duration:  0
value = 0 = 0x0
```

## genToneOff chNum

Syntax:
**`genToneOff chNum`**

Use this command on the VGMC to stop generating a tone started by the genToneOn command.

The following example shows command output.

```
-> genToneOff
value = 0 = 0x0
->
```

## gg_spy_table_show

Syntax:

`gg_spy_table_show`

Display the current Xspy settings.

The following example shows command output where the trace level is modified by the XspySetLevel command.

```
-> gg_spy_table_show
Key Level Destination
----- ---------------- ------------------------ ------------------------
( 1) ROOT dbgPort
( 2) DIM dbgPort
( 3) DIM dbgPort
value = 2794928 = 0x2aa5b0 = _gg_tune_verify_params + 0x148
->
-> XspySetLevel 0,0
value = 0 = 0x0
->
Key Level Destination
----- ---------------- ------------------------ ------------------------
( 1) ROOT General Information dbgPort
( 2) DIM General Information dbgPort
( 3) DIM General Information dbgPort
value = 2389352 = 0x247568 = _gg_tune_verify_params + 0x148
->
-> XspySetLevel 0,2
value = 0 = 0x0
->
-> gg_spy_table_show
Key Level Destination
----- ---------------- ------------------------ ------------------------
( 1) ROOT Normal Event dbgPort
( 2) DIM Normal Event dbgPort
( 3) DIM Normal Event dbgPort
value = 2794928 = 0x2aa5b0 = _gg_tune_verify_params + 0x148
-> XspySetLevel 1,6
value = 0 = 0x0
-> gg_spy_table_show
Key Level Destination
----- ---------------- ------------------------ ------------------------
( 1) ROOT dbgPort
( 2) DIM Normal Event dbgPort
( 3) DIM Normal Event dbgPort
value = 2794928 = 0x2aa5b0 = _gg_tune_verify_params + 0x148
```

### h

Syntax:
`h`

Displays the last 20 commands entered at the VxWorks shell prompt. Prior commands can be recalled, edited and executed, speeding debugging when the same or similar commands must be entered repeatedly.

Press Esc at the VxWorks shell to switch the shell to edit history mode. Press Return to display the line to the shell and exit edit mode. The default value for n is 1. The following are a few of the more useful editing commands.

**Table 29**
**Movement and searching commands**

| Command | Description |
|---|---|
| nG | Go to command |
| nk | Get the nth previous shell command in history. Just entering k returns the last command entered |
| /s | Ssearch for string s backward in history |
| ?s | Search for string s forward in history |
| nh | Move left n characters |
| nl (or n + SPACE) | Move right n characters |
| nw | Move n words forward |
| nb | Move n words back |
| fc | Find character c, searching forward |
| Fc | Find character c, searching backward |
| $ | Go to end of line |
| 0 | Go to start of line |

**Table 30**
**Insertion commands**

| Command | Description |
|---|---|
| a | Append |
| A | Append at end of line |
| cl (or c + SPACE) | Change character (deletes character and enters input mode) |
| cw | Change word (deletes word and enters input mode) |
| cc (or S) | Change entire line |

**Table 30**
**Insertion commands (cont'd.)**

| Command | Description |
|---|---|
| c$ (or C) | Change everything from cursor to end of line |
| i | Insert |
| I | Insert at beginning of line |
| R | Type over characters |

**Table 31**
**Editing commands**

| Command | Description |
|---|---|
| nrc | Replace the following n characters with c. |
| nx | Delete n characters starting at the cursor |
| nX | Delete n characters to the left of the cursor |
| dl | Delete character |
| dw | Delete word |
| dd | Delete entire line |
| d$ (or D) | Delete everything from cursor to end of line |
| p | Put last deletion after theP cursor |
| P | Put last deletion before the cursor |
| u | Undo last command |
| Ctrl+U | Delete line and exit edit mode |
| Ctrl+L | Redraw line |
| Ctrl+D | Complete symbol name |
| RETURN | Give line to shell and exit edit mode |

```
-> h
43 inetstatShow
44 tcpShow
45 udpstatShow
46 itgCardShow
47 dosFsConfigShow
48 mbufShow
49 vgwShow
50 tpsShow
51 i
52 logShow
53 h
54 ls
55 ll
56 pwd
```

```
57 h
58 cd "/C:/CONFIG"
59 copy "CONFIG.INI"
60 h
61 copy "BOOTP.TAB"
62 h value = 0 = 0x0
```

## H323CallTrace ch on

Syntax:

**`H323CallTrace ch on`**

Turns on tracing for all channels.

The following example shows command output.

```
oam> H323CallTrace ch on
oam>
oam>
oam> 11/01/05 15:41:54 LOG0006 NPM: H323CallTrace:  Recv
chid:1 calling:4500 called:4801 remote IP:192.168.1
9.50(1720) Q931 setup 11/01/05 15:41:54 LOG0006 NPM:
H323CallTrace:  Send chid:1 calling:4500 called:4801
remote IP:192.168.19.50(1720) Q931 callProceeding
11/01/05 15:41:54 LOG0006 NPM: H323CallTrace:  Send
chid:1 calling:4500 called:4801 remote IP:192.168.19.
50(1720) Q931 alerting 11/01/05 15:41:56 LOG0006 NPM:
H323CallTrace:  Send chid:1 calling:4500 called:4801
remote IP:192.168.19.50(1720) Q931 connect 11/01/05
15:41:56 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500 called:4801 remote IP:192.168.19.50(1720)
H245 terminalCapabilitySet 11/01/05 15:41:56 LOG0006
NPM: H323CallTrace:  Send chid:1 calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
masterSlaveDetermination 11/01/05 15:41:56 LOG0006 NPM:
H323CallTrace:  Recv chid:1 calling:4500 called:4801
remote IP:192.168.19.50(1720) H245 terminalCapabilitySet
11/01/05 15:41:56 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500 called:4801 remote IP:192.168.19.50(1720)
H245 terminalCapabilitySetAck

oam> 11/01/05 15:41:56 LOG0006 NPM: H323CallTrace:
Recv chid:1 calling:4500 called:4801 remote IP:192.16
8.19.50(1720) H245 masterSlaveDetermination 11/01/05
15:41:56 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500 called:4801 remote IP:192.168.19.50(1720)
H245 masterSlaveDeterminationAck 11/01/05 15:41:56
LOG0006 NPM: H323CallTrace:  Recv chid:1 calling:4500
```

```
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:41:56 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500 called:4801 remote IP:192.168.19.50(1720)
Q931 facility 11/01/05 15:41:56 LOG0006 NPM: H323Cal
lTrace:  Recv chid:1 calling:4500 called:4801 remote
IP:192.168.19.50(1720) H245 terminalCapabilitySetAck
11/01/05 15:41:56 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500 called:4801 remote IP:192.168.19.50(1720)
H245 masterSlaveDeterminationAck 11/01/05 15:41:56
LOG0006 NPM: H323CallTrace:  Recv chid:1 calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:41:58 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500 called:4801 remote IP:192.168.19.50(1720)
H245 closeLogicalChannel 11/01/05 15:41:58 LOG0006 NPM:
H323CallTrace:  Send chid:1 calling:4500 called:4801
remote IP:192.168.19.50(1720) Q931 facility 11/01/05
15:41:58 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500 called:4801 remote IP:192.168.19.50(1720)
H245 terminalCapabilitySet 11/01/05 15:41:58 LOG0006 NPM:
H323CallTrace:  Send chid:1 calling:4500 called:4801
remote IP:192.168.19.50(1720) Q931 facility 11/01/05
15:41:58 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500 called:4801 remote IP:192.168.19.50(1720)
H245 closeLogicalChannel
```

### H323CallTrace ch off

Syntax:

**`H323CallTrace ch off`**

Turns off the tracing for all channels.

### H323CallTrace ch channelNum MsgRecv MsgRecv

Syntax:

**`H323CallTrace ch channelNum MsgRecv MsgRecv`**

Turns the H323 tracing on or off.

The following table describes the command parameters.

**Table 32**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| channelNum | 0 to maximum channel number | Channel number of the virtual trunk to trace. |

**Table 32**
**Command parameters (cont'd.)**

| Parameter | Value | Description |
|---|---|---|
| MsgRecv | ON, OFF | Enables or disables tracing on messages sent to the specified channels. |
| MsgSend | ON, OFF | Enables or disables tracing on messages sent from the specified channels. |

The following example shows command output.

```
oam> H323CallTrace ch 01 on on
oam>
oam>
oam> 11/01/05 15:45:25 LOG0006 NPM: H323CallTrace:  Recv
chid:1 calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 setup
11/01/05 15:45:25 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931
callProceeding
11/01/05 15:45:25 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 alerting
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 connect
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
masterSlaveDetermination
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
masterSlaveDetermination
```

```
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
masterSlaveDeterminationAck
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
masterSlaveDeterminationAck
11/01/05 15:45:27 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
closeLogicalChannel
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
closeLogicalChannel
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
requestChannelClose
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Send chid:1
```

```
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
endSessionCommand
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931
releaseComplete
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
closeLogicalChannelAck
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
closeLogicalChannel
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
endSessionCommand
11/01/05 15:45:28 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
```

### H323CallTrace ch

Syntax:

**`H323CallTrace ch start_chNum end_chNum MsgRecv MsgSend>`**

Enable the tracing of a range of virtual trunk channels.

The following table describes the command parameters.

**Table 33**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| start_chNum | 0 to maximum channel number | First channel number in the range of channels to to trace. |
| end_chNum | 0 to maximum channel number | Last channel number in the range of channels to trace. Must be greater than start_chNum. |
| MsgRecv | ON, OFF | Enables or disables tracing on messages sent to the specified channels. |
| MsgSend | ON, OFF | Enables or disables tracing on messages sent from the specified channels. |

The following example shows command output.

```
oam> H323CallTrace ch 01 06 on on
oam>
oam> 11/01/05 15:46:02 LOG0006 NPM: H323CallTrace:  Recv
chid:1 calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 setup
11/01/05 15:46:02 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931
callProceeding
11/01/05 15:46:02 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 alerting
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 connect
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
masterSlaveDetermination
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Recv chid:1
```

```
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
masterSlaveDetermination
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
masterSlaveDeterminationAck
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
masterSlaveDeterminationAck
11/01/05 15:46:04 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
closeLogicalChannel
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
closeLogicalChannel
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
```

```
called:4801 remote IP:192.168.19.50(1720) H245
requestChannelClose
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
endSessionCommand
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Send chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931
releaseComplete
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
closeLogicalChannelAck
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
closeLogicalChannel
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) H245
endSessionCommand
11/01/05 15:46:05 LOG0006 NPM: H323CallTrace:  Recv chid:1
calling:4500
called:4801 remote IP:192.168.19.50(1720) Q931 facility
oam>
11/01/05 15:46:14 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 setup
11/01/05 15:46:14 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
```

```
called:4500 remote IP:192.168.19.50(1720) Q931
callProceeding
11/01/05 15:46:14 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 alerting
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 connect
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
masterSlaveDetermination
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
masterSlaveDetermination
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
masterSlaveDeterminationAck
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
masterSlaveDeterminationAck
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:17 LOG0006 NPM: H323CallTrace:  Send chid:6
```

```
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
oam> 11/01/05
15:46:21 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
closeLogicalChannel
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
closeLogicalChannel
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
requestChannelClose
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
endSessionCommand
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931
releaseComplete
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
closeLogicalChannel
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Recv chid:6
```

```
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
closeLogicalChannelAck
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
endSessionCommand
11/01/05 15:46:21 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
```

### H323CallTrace num

Syntax:

**`H323CallTrace num calling_number MsgRecv MsgSend`**

Enables the tracing of H.323 messages by using the called and calling numbers. If the called or calling number of a virtual trunk session matches the number specified, the messages to and from the virtual trunk are traced.

The following table describes the command parameters.

**Table 34**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| calling_number | – | Calling or called telephone number to trace. Can be 1–32 numeric digits and can be a partial calling or called number. |
| MsgRecv | ON, OFF | Enables or disables tracing on messages sent to the specified channels. |
| MsgSend | ON, OFF | Enables or disables tracing on messages sent from the specified channels. |

The following example shows command output.

```
oam> H323CallTrace num 4500 on on
oam>
oam> 11/01/05 15:46:39 LOG0006 NPM: H323CallTrace:  Send
chid:6 calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 setup
11/01/05 15:46:39 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931
callProceeding
11/01/05 15:46:40 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 alerting
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 connect
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
masterSlaveDetermination
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
masterSlaveDetermination
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
masterSlaveDeterminationAck
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Recv chid:6
```

```
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
masterSlaveDeterminationAck
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:42 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
closeLogicalChannel IP:192.168.19.50(1720) H245
closeLogicalChannelAck
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySet
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
closeLogicalChannel
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
terminalCapabilitySetAck
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
closeLogicalChannel
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
```

```
called:4500 remote IP:192.168.19.50(1720) H245
requestChannelClose
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
endSessionCommand
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) H245
endSessionCommand
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Send chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931 facility
11/01/05 15:46:43 LOG0006 NPM: H323CallTrace:  Recv chid:6
calling:4801
called:4500 remote IP:192.168.19.50(1720) Q931
releaseComplete
```

### H323CallTrace num calling/called_number NPI TON MsgRecv MsgSend

Syntax:

Enables tracing of H.323 messages using the called and calling numbers. If the called or calling number of a virtual trunk session matches the number specified and the specified NPI and TON values match the call type, then the messages to and from the virtual trunk are traced.

The following table describes the command parameters.

**Table 35**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| channelNum | 0–maximum channel number | Channel number to trace. |

**Table 35**
**Command parameters (cont'd.)**

| Parameter | Value | Description |
|---|---|---|
| NPI | 0–7 | Specify the numbering plan identifier for which to trace calls.<br>0 = ALL NPIs<br>1 = Uknown<br>2 = ISDN/telephone numbering plan (E.164)<br>3 = Private numbering plan<br>4 = E.163<br>5 = Telex numbering plan<br>6 = Data numbering plan<br>7 = National standard numbering plan |
| TON | 0–7 | Specify the type of number to use as a filter for tracing. Only calls using this TON setting are traced.<br>0 = All TONs<br>1 = Unknown Number<br>2 = International Number<br>3 = National Number<br>4 = Network Specific Number<br>5 = Subscriber Number<br>6 = L1 Regional Number<br>7 = L0 Regional Number |
| MsgRecv | ON, OFF | Enables or disables tracing on messages sent to the specified channels. |
| MsgSend | ON, OFF | Enables or disables tracing on messages sent from the specified channels. |

### H323GwShow

Syntax:

**`H323GwShow`**

Prints a snapshot summary of the state of the virtual trunk settings.

The following example shows command output.

```
oam> H323GwShow
Npm status:  Active
Active GateKeeper:  192.168.19.51 (primary)
GateKeeper registration status:  registered, TTL: 25 secs,
re-register:  12 secs
Channels Busy / Idle / Total:  0 / 6 / 6
Stack version:  RadVision 4.1.0.19 Channel tracing:  -1
Signaling Server H323 ID : SS_N318
```

## H323GwShow ch channelNum

Syntax:

**`H323GwShow ch channelNum`**

Display a snapshot summary of the state of the virtual trunk settings plus the snapshot of the active call on the specified channel if the call exists.

The following table describes the command parameters.

**Table 36**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| channelNum | 0–maximum channel number | Channel number to trace. |

The following example shows command output.

```
oam> H323GwShow ch 01
Npm status:  Active
Active GateKeeper:  192.168.19.51 (primary)
GateKeeper registration status:  registered, TTL: 25 secs,
re-register:  19 secs
Channels Busy / Idle / Total:  1 / 5 / 6
Stack version:  RadVision 4.1.0.19 Channel tracing:  -1
Signaling Server H323 ID : SS_N318
Chan Direction CallState RxState TxState Codec AirTime FS
MS Fax DestNum RemoteIP
---- --------- --------- --------- --------- ------------
-------- ------- --- -- --- ------- ---------------
1 Terminate Connected Connected Connected G_711_u_law_20M
S_NOVAD 18 yes m no 4801 192.168.19.50
```

## H323GwShow num calling_num

Syntax:

**`H323GwShow num calling_num`**

Display a snapshot summary of the state of the virtual trunk settings plus the snapshot of the active calls using the calling or called number or partial number specified.

The following table describes the command parameters.

**Table 37**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| calling_num | 0–maximum channel number | Telephone number to trace. Can be 1–32 numeric digits and can be a partial calling or called number. |

The following example shows command output.

```
oam> H323GwShow num 4500
Npm status:  Active
Active GateKeeper:  192.168.19.51 (primary)
GateKeeper registration status:  registered, TTL: 25 secs,
re-register:  14 secs
Channels Busy / Idle / Total:  0 / 6 / 6
Stack version:  RadVision 4.1.0.19
Channel tracing:  -1
Signaling Server H323 ID : SS_N318

Calling/Called Party Number:  4500
Numbering Plan Indicator:  Undefined
Type Of Number:  Undefined
No active calls for the number:  4500, NPI: Undefined, TON:
Undefined
```

### H323GwShow num calling/called_num NPI TON

Syntax:
**`H323GwShow num calling_num NPI TON`**

Display a snapshot summary of the state of the virtual trunk settings plus the snapshot of the active calls using the calling or called number or partial number with the specified NPI and TON values.

The following table describes the command parameters.

**Table 38**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| calling_num | – | Telephone number to trace. Can be 1–32 numeric digits and can be a partial calling or called number. |

**Table 38**
**Command parameters (cont'd.)**

| Parameter | Value | Description |
|---|---|---|
| NPI | 0–7 | Specify the numbering plan identifier for which to trace calls.<br>0 = ALL NPIs<br>1 = Uknown<br>2 = ISDN/telephone numbering plan (E.164)<br>3 = Private numbering plan<br>4 = E.163<br>5 = Telex numbering plan<br>6 = Data numbering plan<br>7 = National standard numbering plan |
| TON | 0–7 | Specify the type of number to use as a filter for tracing. Only calls using this TON setting will be traced.<br>0 = All TONs<br>1 = Unknown Number<br>2 = International Number<br>3 = National Number<br>4 = Network Specific Number<br>5 = Subscriber Number<br>6 = L1 Regional Number<br>7 = L0 Regional Number |

The following example shows command output.

```
oam> H323GwShow num 4500 3 7
Npm status:  Active
Active GateKeeper:  192.168.19.51 (primary)
GateKeeper registration status:  registered, TTL: 25 secs,
re-register:  3 secs
Channels Busy / Idle / Total:  1 / 5 / 6
Stack version:  RadVision 4.1.0.19
Channel tracing:  -1
Signaling Server H323 ID : SS_N318
Calling/Called Party Number:  4500
Numbering Plan Indicator:  Private
Type Of Number:  L0Regional
Chan Direction CallState RxState TxState Codec AirTime FS
MS Fax
DestNum RemoteIP
---- --------- --------- --------- --------- ------------
--------- ------- --- -- --- ------- ---------------
6 Originate Connected Connected Connected G_711_u_law_20M
S_NOVAD 18 yes s no 4500 192.168.19.50
```

### H323Output

Syntax:

**`H323Output output_destination "file_pathname"`**

Specify where to direct tracing output.

The following table describes the command parameters.

**Table 39**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| output_destination | 1–4 | Specify where to direct trace messages for the H323CallTrace command, where:<br>1= TTY<br>2 = RPTLOG<br>3 = File<br>4 = File and TTY |
| file_pathname | "string" | Specify the file to print to if output_destination = 3 or 4. Enclose the string in quotation marks. |

The following example shows command output.

```
oam> H323Output 3 "Testcap.txt"
```

### H323TraceShow

Syntax:

**`H323TraceShow`**

This command displays the trace settings, including the output destination and file name as well as all active traces for the H323CallTrace trace tool.

The following example shows command output.

```
oam> H323TraceShow

Output to TTY

Calling/called number NPI TON H323MsgRecv H323MsgSend
===================== === === =========== ===========
4500 0 0 ON OFF
Channels H323MsgRecv (VTRK->NPM) H323MsgSend (NPM->VTRK)
======== ====================== =======================
0 - 382 OFF OFF
```

### help

Syntax:

**`help`**

Prints the VxWorks shell help menu. The operating system implements these commands on the ITG-P, the SMC, and the Signaling Server.

The following example shows command output.

```
-> help
help Print this list
dbgHelp Print debugger help info
nfsHelp Print nfs help info
netHelp Print network help info
spyHelp Print task histogrammer help info
timexHelp Print execution timer help info
h [n] Print (or set) shell history
i [task] Summary of tasks' TCBs
ti task Complete info on TCB for task
sp adr,args...  Spawn a task, pri=100, opt=0,
stk=20000
taskSpawn name,pri,opt,stk,adr,args...  Spawn a task
td task Delete a task
ts task Suspend a task
tr task Resume a task
d [adr[,nunits[,width]]] Display memory
m adr[,width] Modify memory
mRegs [reg[,task]] Modify a task's registers interactively
pc [task] Return task's program counter
version Print VxWorks version info, and boot
line
iam "user"[,"passwd"] Set user name and passwd
whoami Print user name
devs List devices
cd "path" Set current working path
pwd Print working path
ls ["path"[,long]] List contents of directory
ll ["path"] List contents of directory - long format
rename "old","new" Change name of file
copy ["in"][,"out"] Copy in file to out file (0 = std in/out)
ld [syms[,noAbort][,"name"]] Load stdin, or file, into
memory (syms = add symbols to table:  -1 = none, 0 = globals,
1 = all)
lkup ["substr"] List symbols in system symbol table
lkAddr address List symbol table entries near address
checkStack [task] List task stack sizes and usage
printErrno value Print the name of a status value
period secs,adr,args...  Spawn task to call function
periodically
repeat n,adr,args...  Spawn task to call function n times
(0=forever)
```

```
diskFormat "device" Format disk
diskInit "device" Initialize file system on disk
squeeze "device" Squeeze free space on RT-11 device

NOTE: Arguments specifying 'task' can be either task ID or
name.
value = 1 = 0x1
```

### hostFileGet

Syntax:

**`hostFileGet type, lsnr, "hostIP", "uid", "pswd", "path", "fname", "dPath"`**

Transfer the specified file from the directory path on the specified FTP server to the specified directory. Neither is used when you manually transfer the file from the VxWorks shell. For examples of user names and passwords for various FTP servers, see "swDownload " (page 258).

**Table 40**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| type | 0 | Define the file type (for example, BOOTP) for file locking. |
| lsnr | 0 | Define the task to notify when the file transfer completes. |
| hostIP | "string" | Specify the FTP server host IP address from which to retrieve the file. Enclose the string in quotation marks. |
| uid | "string" | Username to log on to the remote host. |
| pswd | "string" | Password to log on to the remote host. |
| fname | "string" | Specify the file name of the file to retrieve. Enclose the string in quotation marks. |
| dPath | "string" | Specify the path name to transfer the file to. Enclose the string in quotation marks. |

The following example shows command output.

```
VGMC> hostFileGet
0,0,"192.168.1.14","itgadmin","itgadmin","/C:","CONFIG.
1","/C:/CONFIG"
value = 0 = 0x0
```

### hostFilePut

Syntax:

```
hostFilePut type,lsnr,"hostIP","uid","pswd","path","f
name","srcFile"
```

Transfer the specified file from the VGMC to the specified host. For examples of user names and passwords for various FTP servers, see "swDownload " (page 258).

**Table 41**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| type | 0 | Define the file type (for example, BOOTP) for file locking. |
| lsnr | 0 | Define the task to notify when the file transfer completes. |
| hostIP | "string" | Specify the FTP server host IP address to transfer the file to. Enclose the string in quotation marks. |
| uid | "string" | Username to log on to the remote host. Enclose the string in quotation marks. |
| pswd | "string" | Password to log on to the remote host. Enclose the string in quotation marks. |
| fname | "string" | Specify the file name to save the file as on the remote host. Enclose the string in quotation marks. |
| path | "string" | Specify the path name to transfer the file to. Enclose the string in quotation marks. |
| srcfile | "string" | Specify the file name of the source file on the VGMC. Enclose the string in quotation marks. |

```
VGMC> hostFilePut
0,0,"192.168.1.14","itgadmin","itgadmin","/C:","CONFIG.
1","CONFIG.INI"

FEB 21 11:08:28 tShell:  Notice File transfer starting:
CONFIG.INI -> 192.168.1.14:/C:/CONFIG.1
FEB 21 11:08:29 tShell:  Notice File transfer completed:
CONFIG.INI ->
192.168.1.14:/C:/CONFIG.1
value = 0 = 0x0
```

### i

Syntax:

```
i
```

Print all tasks and task IDs to ensure that no task is suspended. For more information about these tasks, see "VGMC and Signaling Server logging commands" (page 57).

```
VGMC> i
NAME ENTRY TID PRI STATUS PC SP ERRNO DELAY
---------- ------------ -------- --- ---------- --------
-------- ------ -----
tExcTask _excTask 3fadc4c 0 PEND 300f7f 3fadbbc 0 0
tShell _shell 3a881f8 1 READY 2e40d8 3a87ed8 3006b 0
tPcmciad _pcmciad 3f8aa24 2 PEND 300f7f 3f8a994 0 0
tTelnetd _telnetd 3aa3cf0 2 PEND 2a9e50 3aa3c24 0 0
tPxTimer _pxTaskInit 3a72ed0 10 DELAY 2a7b26 3a72e6c 4
29917
tRdbTask _rdbTask 3aa1290 20 PEND 2a9e50 3aa117c d0003 0
tMVX_DIM _dim_main 3a4e488 40 PEND 2a9e50 3a4e41c 3d0002 0
tRTP _vgwRtpTask 3a14814 40 PEND 2a9e50 3a145c4 0 0
tAioIoTask1_aioIoTask 3f9cd48 50 PEND 2a9e50 3f9ccf0 0 0
tAioIoTask0_aioIoTask 3f95ba0 50 PEND 2a9e50 3f95b48 0 0
tNetTask _netTask 3c34188 50 PEND 2a9e50 3c34130 3d 0
tAioWait _aioWaitTask 3fa3ef0 51 PEND 2a9e50 3fa3dfc 0 0
tFtpdTask _ftpdTask 3a9eb68 55 PEND 2a9e50 3a9eaa0 0 0
tTftpdTask _tftpdTask 3a9b9ac 55 PEND 2a9e50 3a9b438 0 0
baseMMintTa_baseMMintTa 3a6b784 70 PEND 2a9e50 3a6b714 0 0
tA07 _a07Task 3a66614 80 PEND 300f7f 3a66598 0 0
tTffsPTask _flPollTask 3f8830c 100 DELAY 2a7b26 3f882c8
3d0002 41
tPortmapd _portmapd 3aa27c0 100 PEND 2a9e50 3aa2698 16 0
tXA _xaTask 3a68ecc 100 PEND 300f7f 3a68dac 0 0
tRDP _rudpMgrStar 3a643e4 120 PEND 2a9e50 3a64178 b 0
tSnmpd 2d056c 3a98900 150 PEND 2a9e50 3a97f84 0 0
tTCK _tpsSocketTa 3a21124 195 PEND 2a9e50 3a20ee8 0 0
tbootpd _cmain 3a6f670 200 PEND 2a9e50 3a6f124 23 0
tMAM _mamMain 3a57288 200 PEND 2d7112 3a571b4 2 0
tVTM 271598 3a39edc 200 PEND 2d7112 3a39e04 4 0 tSET 294e88
3a2d684 200 PEND 2d7112 3a2d5a8 380003 0
tCSV _csvTask 3a23bd4 200 PEND 2d7112 3a23b00 4 0
tTPS _tpsTask 3a1ee3c 200 PEND 2d7112 3a1ed38 4 0
tVGW _vgwTask 3a169bc 200 PEND 2d7112 3a168e0 d0003 0
tMVX_XSPY _XSpyTaskMai 3a50a14 250 PEND 300f7f 3a50984
c0002 0
tOMM _ommMain 3a3dbf4 250 PEND 2d7112 3a3db20 1c0001 0
tVTI _vtiTask 3a30afc 250 PEND 2d7112 3a30a2c 1c0001 0
tUMS _umsServerSt 3a1c2c4 250 PEND 2d7112 3a1c1dc 4 0
tUMC _umsClientSt 3a19780 250 PEND 2d7112 3a196a0 1c0001 0
tLogTask _logTask 3fa9ee4 255 PEND 300f7f 3fa9e58 0 0
```

```
tSyslogd 115bf8 3a75078 255 PEND 300f7f 3a74e58 0 0
value = 0 = 0x0
VGMC>
```

### icmpstatShow

Syntax:
**`icmpstatShow`**

Displays statistics for the ICMP protocol.

```
-> icmpstatShow ICMP: 4 calls to icmp_error 0 error not
generated because old message was icmp Output histogram:
echo reply:  7 destination unreachable:  4 0 message with
bad code fields 0 message < minimum length 0 bad checksum 0
message with bad length Input histogram:  echo reply:  9
destination unreachable:  1 echo:  7 7 message responses
generated value = 31 = 0x1f
```

### ifShow

Syntax:
**`ifShow`**

Displays the attached network interfaces. The current parameters configured on the ELAN (lnIsa) and TLAN (lnPci) are printed. On the current Master card, two Internet addresses print for the lnPci interface: the first is the card TLAN interface IP, and the second is the node IP.

The lo parameter is the internal software loopback interface. The Ethernet address (MAC address) prints for each interface. A large number of multicast packets received indicates extensive broadcast traffic on the interface, which is detrimental to the performance of the VGMC. If no lnPci interface data prints, the card did not boot up correctly and the interface is not configured. The card management MAC address may not match the MAC address configured in EM/OTM; if that is true, correct it in EM/OTM, download the node properties from OTM (or submit and transfer in EM) and reboot the card.

The following is the output on the ITG-P card.

```
VGMC> ifShow lnIsa (unit number 0):
Flags:  (0x8863) UP BROADCAST MULTICAST ARP RUNNING
Type:  ETHERNET_CSMACD
Internet address:  47.11.216.181
Broadcast address:  47.11.217.255
Netmask 0xff000000 Subnetmask 0xfffffe00
Ethernet address is 00:60:38:8e:29:b9
Metric is 0
```

```
Maximum Transfer Unit size is 1500
95696 packets received; 1848 packets sent
93332 multicast packets received
0 multicast packets sent
0 input errors; 0 output errors
0 collisions; 0 dropped
lnPci (unit number 1):
Flags:  (0x8863) UP BROADCAST MULTICAST ARP RUNNING
Type:  ETHERNET_CSMACD
Internet address:  47.11.215.159
Broadcast address:  47.11.215.255
Netmask 0xff000000 Subnetmask 0xffffff00
Ethernet address is 00:60:38:bd:20:92
Metric is 0
Maximum Transfer Unit size is 1500
96270 packets received; 1723 packets sent
93319 multicast packets received
0 multicast packets sent
0 input errors; 0 output errors
0 collisions; 0 dropped lo (unit number 0):
Flags:  (0x8069) UP LOOPBACK MULTICAST ARP RUNNING
Type:  SOFTWARE_LOOPBACK
Internet address:  127.0.0.1
Netmask 0xff000000 Subnetmask 0xff000000
Metric is 0
Maximum Transfer Unit size is 32768
90 packets received; 90 packets sent
0 multicast packets received
0 multicast packets sent
0 input errors; 0 output errors
0 collisions; 0 dropped
value = 29 = 0x1d
```

The following is the output on the SMC card.

```
VGMC> ifShow ixpMac (unit number 1):
Flags:  (0x8863) UP BROADCAST MULTICAST ARP RUNNING
Type:  ETHERNET_CSMACD
Internet address:  47.11.216.246
Broadcast address:  47.11.217.255
Netmask 0xff000000 Subnetmask 0xfffffe00
Ethernet address is 00:60:38:bd:b3:51
Metric is 0
Maximum Transfer Unit size is 1500
102786 packets received; 1417 packets sent
101405 multicast packets received
0 multicast packets sent
```

```
0 input errors; 0 output errors
0 collisions; 0 dropped
lo (unit number 0):
Flags:  (0x8069) UP LOOPBACK MULTICAST ARP RUNNING
Type:  SOFTWARE_LOOPBACK
Internet address:  127.0.0.1
Netmask 0xff000000 Subnetmask 0xff000000
Metric is 0
Maximum Transfer Unit size is 32768
0 packets received; 0 packets sent
0 multicast packets received
0 multicast packets sent
0 input errors; 0 output errors
0 collisions; 0 dropped
ixpMac (unit number 0):
Flags:  (0x8863) UP BROADCAST MULTICAST ARP RUNNING
Type:  ETHERNET_CSMACD
Internet address:  47.11.215.30
Broadcast address:  47.11.215.255
Netmask 0xff000000 Subnetmask 0xffffff00
Ethernet address is 00:60:38:bd:b3:50
Metric is 0
Maximum Transfer Unit size is 1500
102855 packets received; 548 packets sent
102478 multicast packets received
40 multicast packets sent
0 input errors; 0 output errors
0 collisions; 0 dropped
value = 29 = 0x1d
```

The following is the command from the Signaling Server. The Signaling Server is currently the node Master; the TLAN interface (fei unit number 1) has a second IP address assigned, which is the node IP address.

```
-> ifShow fei (unit number 0):
Flags:  (0x8063) UP BROADCAST MULTICAST ARP RUNNING
Type:  ETHERNET_CSMACD
Internet address:  47.11.217.158
Broadcast address:  47.11.217.255
Netmask 0xff000000 Subnetmask 0xfffffe00
Ethernet address is 00:03:47:da:cd:59
Metric is 0
Maximum Transfer Unit size is 1500
16968943 octets received
3912683 octets sent
175445 packets received
36349 packets sent
```

```
143228 broadcast packets received
56 broadcast packets sent
0 multicast packets received
0 multicast packets sent
0 input discards
0 input unknown protocols
0 input errors
0 output errors
0 collisions; 0 dropped
lo (unit number 0):
Flags:  (0x8069) UP LOOPBACK MULTICAST ARP RUNNING
Type:  SOFTWARE_LOOPBACK
Internet address:  127.0.0.1

Internet address:  47.11.215.43
Netmask 0xff000000 Subnetmask 0xff000000
Metric is 0
Maximum Transfer Unit size is 32768
826 packets received; 826 packets sent
0 multicast packets received
0 multicast packets sent
0 input errors; 0 output errors
0 collisions; 0 dropped fei (unit number 1):
Flags:  (0x8063) UP BROADCAST MULTICAST ARP RUNNING Type:
ETHERNET_CSMACD
Internet address:  47.11.215.44
Broadcast address:  47.11.215.255
Internet address:  47.11.215.43
Broadcast address:  47.11.215.255
Netmask 0xff000000 Subnetmask 0xffffff00
Ethernet address is 00:03:47:da:cd:5a
Metric is 0
Maximum Transfer Unit size is 1500
13868553 octets received
9164036 octets sent
152939 packets received
13719 packets sent
142752 broadcast packets received
532 broadcast packets sent
0 multicast packets received
0 multicast packets sent
0 input discards
0 input unknown protocols
0 input errors
0 output errors
0 collisions; 0 dropped
value = 29 = 0x1d
```

### inactiveDlogShow numOfLine

Syntax:

**`inactiveDlogShow numOfLine`**

Show the nonactive log file information for the UFTP IP Telephone firmware download. When no argument is passed, the command shows the contents of the entire file. When you enter the optional parameter numOfLine, the command prints the number of lines specified.

```
oam> inactiveDlogShow inactiveDlogShow Active F/W download
file:  /u/log/UFTPLOG0.TXT Space remaining:  399755
Inactive F/W download file:  /u/log/UFTPLOG1.TXT /u/log/
UFTPLOG1.TXT ------------------------------------------
12/27/03 19:58:41 f/w dnld success:  (47.11.217.11) I2002
12/27/03 20:24:30 f/w dnld success:  (47.11.217.12) I2002
12/27/03 21:42:11 f/w dnld success:  (47.11.217.15) I2002
12/27/03 22:17:40 f/w dnld success:  (47.11.217.20) I2004
```

### inetstatShow

Syntax:

**`inetstatShow`**

Display information about all of the active IP sockets.

```
-> inetstatShow Active Internet connections (including
servers) PCB Proto Recv-Q Send-Q Local Address
Foreign Address (state) -------- ----- ------ ------
------------------ ------------------ -------
3c9454c TCP 0 0 192.168.1.14.23 192.168.1.102.1672
ESTABLISHED 3c93e98 TCP 0 0 0.0.0.0.111 0.0.0.0.0 LISTEN
3c93d0c TCP 0 0 0.0.0.0.21 0.0.0.0.0 LISTEN 3c93c04 TCP 0 0
0.0.0.0.1009 0.0.0.0.0 LISTEN 3c93afc TCP 0 0 0.0.0.0.23
0.0.0.0.0 LISTEN 3c94444 UDP 0 0 192.168.1.149.4100
0.0.0.0.0 3c943c0 UDP 0 0 0.0.0.0.16543 0.0.0.0.0 3c9433c
UDP 0 0 192.168.1.140.7300 0.0.0.0.0 3c942b8 UDP 0 0
192.168.1.14.15000 0.0.0.0.0 3c941b0 UDP 0 0 0.0.0.0.15001
0.0.0.0.0 3c93c88 UDP 0 0 192.168.1.140.5100 0.0.0.0.0
3c940a8 UDP 0 0 0.0.0.0.514 0.0.0.0.0 3c94024 UDP 0 0
0.0.0.0.20001 0.0.0.0.0 3c93fa0 UDP 0 0 0.0.0.0.67
0.0.0.0.0 3c93f1c UDP 0 0 0.0.0.0.161 0.0.0.0.0 3c93e14
UDP 0 0 0.0.0.0.111 0.0.0.0.0 3c93d90 UDP 0 0 0.0.0.0.69
0.0.0.0.0 value = 1 = 0x1 ->
```

### iosFdShow

Syntax:

**`iosFdShow`**

Display all of the file descriptors in use.

The following example is from an ITG-P card.

```
-> iosFdShow
fd name drv
3 /tyCo/0 1
4 /aioPipe
2 5 (socket)
4 6 (socket)
4 7 (socket)
4 8 /dev/log
2 9 (socket)
4 10 (socket)
4 11 (socket)
4 12 (socket)
4 13 /dev/log
2 14 /C:/log/EXCPLOG.0
3 15 /C:/log/audit.his
3 16 /C:/log/SYSLOG.0
3 17 (socket)
4 18 /pipe/bootpd
2 19 (socket)
4 20 /dev/log
2 21 (socket)
4 22 /dev/log
2 23 /pipe/srv.6
2 24 /dev/log
2 25 /dev/log
2 26 /pipe/rudp
2 27 /dev/log
2 28 (socket)
4 29 /dev/log
2 30 /dev/log
2 31 (socket)
4 32 (socket)
4 33 (socket)
4 34 (socket)
4 35 /dev/log
2 36 /dev/log
2 37 /dev/log
2 38 (socket)
4 39 /pipe/srv.39
2 40 /dev/log
2 41 /dev/log
2 42 /dev/log
2 43 /dev/log
```

```
2 44 /dev/log
2 45 (socket)
4 46 (socket)
4 47 /pipe/srv.38
2 48 /dev/log
2 49 /dev/log
2 50 /pty/telnet.M
7 51 /dev/log
2 52 /dev/log
2 53 (socket)
4 54 /dev/log
2 55 /pipe/rtpsig
2 56 /dev/log
2 57 /pipe/rtcpsig
2 58 /dev/log
2 59 /dev/log
2 60 (socket)
4 61 /pty/telnet.S 6 in out err
62 /tyCo/0
1 63 /dev/log
2 value = 33286644 = 0x1fbe9f4
->
```

The following example is the output on the SMC.

```
-> iosFdShow
fd name drv
3 /tyCo/0 1
4 /aioPipe 2
5 (socket) 4
6 (socket) 4
7 (socket) 4
8 (socket) 4
9 (socket) 4
10 (socket) 4
11 (socket) 4
12 /C:/log/excplog.0 3 << Fd for active Exception Log file
13 /C:/log/audit.his 3 << Fd for audit history file
14 (socket) 4
15 /pipe/bootpd 2 << Fd for Bootp 16 /dev/log 2
17 (socket) 4
18 /C:/log/syslog.2 3 << Fd for active SYSLOG file
19 /dev/log 2
20 /pipe/srv.6 2
21 /dev/log 2
22 /dev/log 2
23 /pipe/rudp 2
```

```
24 /dev/log 2
25 /dev/log 2
26 (socket) 4
27 (socket) 4
28 (socket) 4
29 (socket) 4
30 (socket) 4
31 /dev/log 2
32 /dev/log 2
33 /dev/log 2
34 /dev/log 2
35 (socket) 4
36 /pipe/srv.39 2
37 /dev/log 2
38 /dev/log 2
39 /dev/log 2
40 /dev/log 2
41 /dev/log 2
42 (socket) 4
43 /dev/log 2
44 (socket) 4
45 (socket) 4
46 /pipe/srv.38 2
47 /dev/log 2
48 /dev/log 2
49 /dev/log 2
50 /dev/log 2
51 /pty/telnet.M 7
52 /pty/telnet.S 6 in out err
53 /dev/log 2
54 /tyCo/0 1
55 /dev/log 2
56 (socket) 4
57 /dev/log 2
58 /pipe/rtpsig 2
59 /dev/log 2
60 /pipe/rtcpsig 2
61 /dev/log 2
value = 32 = 0x20 = ' '
```

The following example is the output on the Signaling Server.

```
-> iosFdShow
fd name drv
3 /tyCo/0 2 in out err
4 /aioPipe 3
5 /tyCo/1 2
```

```
6 (socket) 8
7 (socket) 8
8 (socket) 8
9 (socket) 8
10 (socket) 8
11 /u/rpt 4
12 /u/rpt/LOG00000.RPT 4
13 /u/rpt 4
14 /u/rpt/LOG00000.RPT 4
15 (socket) 8
16 (socket) 8
17 (socket) 8
18 (socket) 8
19 /pipe/bootpd 3
20 /pipe/srv.6 3
21 (socket) 8
22 /pipe/rudp 3
23 (socket) 8
24 (socket) 8
25 (socket) 8
26 (socket) 8
27 (socket) 8
28 /pipe/srv.39 3
29 (socket) 8
30 (socket) 8
31 (socket) 8
32 /pipe/srv.38 3
33 (socket) 8
34 (socket) 8
35 /pipe/srv.48 3
36 (socket) 8
37 (socket) 8
39 (socket) 8
40 /pipe/srv.49 3
41 (socket) 8
42 (socket) 8
43 (socket) 8
45 (socket) 8
value = 32752 = 0x7ff0
```

### IPInfoShow

Syntax:

**`IPInfoShow`**

Display a summary of the card IP configuration from the VGMC prompt.

The following is the output from the ITG-P card.

```
VGMC> IPInfoShow
Maintenance Interface = lnIsa0
Maintenance IP address = 192.168.1.14
Maintenance subnet mask = 255.255.255.128
Voice Interface = lnPci1
Voice IP address = 192.168.1.140
Voice subnet mask = 255.255.255.128

ROUTE NET TABLE destination gateway flags Refcnt Use
Interface ---------------------------------------------
------------------------------
0.0.0.0 192.168.1.200 3 0 0 lnPci1
192.168.1.0 192.168.1.14 101 0 0 lnIsa0
192.168.1.128 192.168.1.140 101 0 0 lnPci1
--------------------------------------------------------
---------------------
ROUTE HOST TABLE destination gateway flags Refcnt Use
Interface ---------------------------------------------
------------------------------ 127.0.0.1 127.0.0.1 5 0 0
lo0 ---------------------------------------------------
------------------------ value = 77 = 0x4d = 'M'
```

The following is the output on the SMC.

```
VGMC> IPInfoShow Maintenance Interface = ixpMac1
Maintenance IP address = 47.11.216.246 Maintenance subnet
mask = 255.255.254.0 Voice Interface = ixpMac0 Voice IP
address = 47.11.215.30 Voice subnet mask = 255.255.255.0
ROUTE NET TABLE destination gateway flags Refcnt Use
Interface ------------------------------------------
--------------------------------- 0.0.0.0 47.11.215.1
3 2 2921 ixpMac0 47.11.215.0 47.11.215.30 101 0 0 ixpMac0
47.11.216.0 47.11.216.246 101 0 0 ixpMac1 47.11.215.0
47.11.215.30 101 0 0 ixpMac0 ----------------------------
------------------------------------------------ ROUTE
HOST TABLE destination gateway flags Refcnt Use Interface
-----------------------------------------------------
------------------------ 127.0.0.1 127.0.0.1 5 0 0 lo0
--------------------------------------------------------
--------------------- value = 77 = 0x4d = 'M'
```

The following is the output on the Signaling Server.

```
-> IPInfoShow Maintenance Interface = fei0 Maintenance
IP address = 47.11.217.158 Maintenance subnet mask =
255.255.254.0 Voice Interface = fei1 Voice IP address
= 47.11.215.44 Voice subnet mask = 255.255.255.0 ROUTE
```

```
NET TABLE destination gateway flags Refcnt Use Interface
---------------------------------------------------
------------------------- 0.0.0.0 47.11.215.1 3 1 756
fei1 47.11.215.0 47.11.215.44 101 0 0 fei1 47.11.216.0
47.11.217.158 101 0 0 fei0 -----------------------------
----------------------------------------------- ROUTE
HOST TABLE destination gateway flags Refcnt Use Interface
-------------------------------------------------------
--------------------- 47.11.215.43 47.11.215.43 5 0 0 lo0
127.0.0.1 127.0.0.1 5 1 19 lo0 ---------------------------
------------------------------------------------- value
= 77 = 0x4d = 'M'
```

## ipstatShow

Syntax:

**`ipstatShow`**

Display the IP protocol statistics.

```
-> ipstatShow
total 1365243 badsum 0 tooshort 0 toosmall 0 badhlen 0
badlen 0 infragments 0 fragdropped 0 fragtimeout 0 forward
0 cantforward 911 redirectsent 0 unknownprotocol 918
nobuffers 0 reassembled 0 outfragments 0 noroute 0

value = 1 = 0x1
```

## isetCount

Syntax:

**`isetCount "expressionString"`**

Count the number of registered IP Phones based on the specified query. This command uses the same parameter as the isetGet command, but instead of printing out detailed phone information, it prints only the total number of telephones that satisfy the query.

The following are examples of various expressions and the numbers returned by isetCount based on the following TPS IP Phone status.

-> isetGet IP Address Type RegType State Up Time TN HWID FWVsn UNIStimVsn SrcPort DstPort
------------------ ------- ------- ----------- ------------- ------------ ------------------- -------
---------- ------- -------
47.11.254.12 i2004 Regular busy 5 16:50:31 061-02 1800603876c7a1660
0 0602B39 2.5 5100 5000
47.11.254.21 i2002 Regular busy 5 16:50:26 061-11 1800802ddcd514660
0 0603B39 2.5 5100 5000

```
47.11.254.11 i2004 Regular online 5 16:49:48 061-01 180060387641c366
00 0602B39 2.5 5100 5000
Total sets = 3
value = 0 = 0x0

-> isetCount Total
Internet Phone Phone Count = 3
value = 0 = 0x0

-> isetCount "ip > 47.11.254.11"
IP Phone Count = 2
value = 0 = 0x0

-> isetCount "ip >= 47.11.254.11"
IP Phone Count = 3
value = 0 = 0x0

-> isetCount "tn != 61 11"
IP Phone Count = 2
value = 0 = 0x0

-> isetCount "state == busy"
IP Phone Count = 2
value = 0 = 0x0

-> isetCount "state == busy && type != i2004"
IP Phone Count = 1
value = 0 = 0x0

-> isetCount "fwvsn >= 0602b30"
IP Phone Count = 3
value = 0 = 0x0

-> isetCount "ip < 47.11.254.12"
IP Phone Count = 1
value = 0 = 0x0

-> isetCount "fwvsn <= 0602b39"
IP Phone Count = 3
value = 0 = 0x0
```

### isetGet

Syntax:

**`isetGet "expressionString"`**

This command uses one string parameter that is a sequence of expression, linked by &&. Each expression consists of three parts: opcode1, operator and opcode2. Currently, only the following operators are handled: ==, !=, <, >, <=, and >=. opcode1 must be one of the fields in the isetShow command output, opcode2 must be a well-formatted value, the format depends on opcode1. Both opcode1 and opcode2 are not case-sensitive.

For use "NAT==x", where x can be:
C for Cone NAT
S for Symmetric NAT
P for Pending a response from Echo Server 1
U for Unknown, response from Echo Server 1, but none from Echo Server 2.

Behind NAT, but unsure if Cone or Symmetric.
Y for all telephones that are "C,S, and U"
N for all telephones that are not behind NAT
<blank> for all telephones that are not behind NAT

"RegType" is not currently accepted as an opcode.

```
"IP == 47.11.216.242"
"tn == 61 23"
"TN == 61 23 && IP == 47.11.216.242"
"Type == i2004 && State == online"
```

If the parameter is "?", then a help text prints.

The output of the command isetGet without parameters is the same as entering isetShow without parameters.

```
-> isetShow IP Address Type RegType State Up Time TN HWID
FWVsn UNIStimVsn SrcPort DstPort
-------------- ------- ------- ----------- -------------
---------- ------------------ ------- ---------- -------
------
47.11.254.12 i2004 Regular online 0 16:50:31 061-02
1800603876c7a16600 0602B39 2.5 5100 5000
47.11.254.21 i2002 Regular busy 0 16:50:26 061-11
1800802ddcd5146600 0603B39 2.5 5100 5000
47.11.254.11 i2004 Regular online 0 16:49:48 061-01
180060387641c36600 0602B39 2.5 5100 5000
Total sets = 3
value = 0 = 0x0

-> isetGet
```

```
IP Address Type RegType State Up Time TN HWID FWVsn
UNIStimVsn SrcPort DstPort
-------------- ------- ------- ----------- -------------
---------- ------------------ ------- ---------- -------
------
47.11.254.12 i2004 Regular online 0 16:50:32 061-02
1800603876c7a16600 0602B39 2.5 5100 5000
47.11.254.21 i2002 Regular busy 0 16:50:27 061-11
1800802ddcd5146600 0603B39 2.5 5100 5000
47.11.254.11 i2004 Regular online 0 16:49:49 061-01
180060387641c36600 0602B39 2.5 5100 5000

Total sets = 3
value = 0 = 0x0
```

## Query Expression

Syntax:

The query string is a sequence of expressions, linked by &&. Each expression consists of three parts: opcode1, operator, and opcode2. Currently, only the following operators are handled: ==, !=, <, >, <=, and >= .

opcode1 must be one of the fields seen in "isetShow" output (except RegType).

opcode2 must be a well-formatted value, the format depends on opcode1.

Both opcode1 and opcode2 are not case-sensitive.

```
"IP == 47.11.216.242"
"tn == 61 23"
"TN == 61 23 && IP == 47.11.216.242"
"Type == i2004 && State == online"
"NAT == y"

2.-> isetGet "ip > 47.11.215.68"
IP Address Type State Up Time TN HWID FWVsn SrcPort DstPort
------------------ ------- ----------- ------------
------------ ------------------ ------- ------- -------
47.11.215.69 i2002 busy 0 00:00:28 061-22
18006038dd00416600 0603B30 5100 5000

Total sets = 1
```

```
3.  -> isetGet "ip >= 47.11.215.68"
IP Address Type State Up Time TN HWID FWVsn SrcPort DstPort
------------------ ------- ----------- ------------
------------ ------------------ ------- ------- -------
47.11.215.68 i2004 busy 0 00:00:34 061-23 180060387602a766
00 0602B28 5100 5000
47.11.215.69 i2002 busy 0 00:00:16 061-22 18006038dd004166
00 0603B30 5100 5000
Total sets = 2

4.  -> isetGet "tn != 61 22"
IP Address Type State Up Time TN HWID FWVsn SrcPort DstPort
------------------ ------- ----------- ------------
------------ ------------------ ------- ------- -------
47.11.215.68 i2004 busy 0 00:02:31 061-23 180060387602a766
00 0602B28 5100 5000
Total sets = 1

5.  -> isetGet "state == busy"
IP Address Type State Up Time TN HWID FWVsn SrcPort DstPort
------------------ ------- ----------- ------------
------------ ------------------ ------- ------- -------
47.11.215.68 i2004 busy 0 00:02:48 061-23 180060387602a766
00 0602B28 5100 5000
47.11.215.69 i2002 busy 0 00:02:30 061-22 18006038dd004166
00 0603B30 5100 5000
Total sets = 2

6.-> isetGet "state == busy && type != i2004"
IP Address Type State Up Time TN HWID FWVsn SrcPort DstPort
------------------ ------- ----------- ------------
------------ ------------------ ------- ------- -------
47.11.215.69 i2002 busy 0 00:03:09 061-22 18006038dd004166
00 0603B30 5100 5000
Total sets = 1

7.-> isetGet "fwvsn >= 0602b30"
IP Address Type State Up Time TN HWID FWVsn SrcPort DstPort
------------------ ------- ----------- ------------
------------ ------------------ ------- ------- -------
47.11.215.69 i2002 busy 0 00:04:27 061-22 18006038dd004166
00 0603B30 5100 5000
Total sets = 1

8.-> isetGet "ip < 47.11.215.69"
IP Address Type State Up Time TN HWID FWVsn SrcPort DstPort
------------------ ------- ----------- ------------
```

```
------------ ------------------ ------- ------- -------
47.11.215.68 i2004 busy 0 00:07:21 061-23 180060387602a766
00 0602B28 5100 5000
Total sets = 1

9.  -> isetGet "fwvsn <= 0602b30"
IP Address Type State Up Time TN HWID FWVsn SrcPort DstPort
------------------ ------- ----------- ------------
------------ ------------------ ------- ------- -------
47.11.215.68 i2004 busy 0 00:08:47 061-23 180060387602a766
00 0602B28 5100 5000
Total sets = 1

10.  -> isetGet "NAT == y"

Set Information
---------------
IP Address NAT Type RegType State Up Time Set-TN Regd-TN
HWID FWVsn UNIStimVsn SrcPort DstPort
---------------- ----- ---------- ------- ----------- ----
---------- ------------ ------------ --------------------
------- ---------- ------ -------
47.11.179.168 C i2004 Regular online 0 04:20:34 061-00
061-00 1800-6038b689e9-6600 0602B59 2.8 5100 5000
47.11.179.167 C i2004 Regular online 0 03:48:17 061-01
061-01 1800-60387602b9-6600 0602B59 2.8 5100 5000
Total sets = 2
```

## isetHlocShow

Syntax:

**isetHlocShow "IPAddr" or "TN"**

Print the PD related settings for an IP Phone (added by the PD/RL/CL feature). Enter either the IP Phone public IP address or TN as the parameter.

The output contains the telephone IP address, home location code (HLOC), Private Network Identifier (PNI), and customer number. Other parameters in the second row are the configured user preferences: preferred name match, log mode, new call indicator, name display format, and area codes.

```
-> isetHlocShow "47.11.215.136"
dsetHlocShowByIP: ip:  47.11.215.136, HLOC: , PNI: 10,
cust_no:  0
pnm:  0, logMode:  0, newCallInd:  1, nameDispFormat:  1,
AC1:, AC2:, AC3:.
```

## isetInfoShow

Syntax:

**`isetInfoShow "IPAddr" or "TN"`**

This command displays standard DHCP configuration information and other telephone information such as firmware version, hardware identification, and server information report. Run this command from the VGMC> prompt.

The following example shows command output.

```
isetInfoShow Report from Set (47.11.215.153)
FW Version:  0602B50
HWID: 18006038DD1ADB6600
MAC: 006038DD1ADB
VLAN ID: 124
Priority:  6
Set IP: 47.103.225.125
Subnet Mask:  255.255.255.0
Set Gateway:  47.103.225.1
LTPS IP: 47.103.247.224
Node IP: 47.103.247.224
Node ID: 4420 S1
Node IP: 47.103.247.229
Port:  4100 Action:  1
S2 Node IP: 47.103.247.229
Port:  4100 Action:  1
S5 Node IP: 47.103.247.229
Port:  4100
XAS: Net6

-> isetInfoShow "47.11.213.216"
value = 247380368 = 0xebeb990
->

isetInfoShow Report (DHCPConfig) from Set (47.11.213.216)

Terminal Type:  i2004 Ph2
Firmware Version:  0604D48
Hardware ID: 18-000ae40acb81-6602
Release Number:  0x02
Manufacture Code:  0x000ae4
Color Code:  0x66
PEC Code:  NTDU92AA
DHCP Server IP: 255.255.255.255
VLAN Priority:  6
VLAN ID: 65535
```

```
Set IP Address:  47.11.213.216
Set Subnet Mask:  255.255.255.0
Set IP Gateway Address:  47.11.213.1
Boot Mode:  11

isetInfoShow Report (Server Info) from Set (47.11.213.216)

Server 1 Server IP = 47.11.239.230
Port Number = 4100
Action = 1
Retry = 10
Server 2 Server IP = 0.0.0.0
Port Number = 0
Action = 1
Retry = 1
Server 3 Server IP = 47.11.239.235
Port Number = 5100
Action = 1
Retry = 0
```

## isetNATShow

Syntax:

**`isetNATShow “IPAddr” or “TN”`**

The isetNATShow addresses the need for additional information about IP Phones on a NAT device. Optionally, you can enter an IP Phone TN or public signaling IP address as a parameter, similar to the isetShow command.

```
VGMC>isetNatShow
signaling Media Public IP Addr:Port Public IP Addr:Port
(Private IP Addr:Port) (Private IP Addr:Port) NAT Type RTCP
Type Set-TN Reg-TN
---------------------- -----------------------
--------- ---- ------ ------------ ------------
47.11.217.28:5284 47.11.217.28:5289 Cone Yes i2004 61-09
61-09
(192.168.0.102:5000) (192.168.0.102:5200) 47.11.222.45
:6200 <No speech possible> Symmetric No i2004 64-03 64-09
(192.168.0.100:5000)
47.12.48.12:4398 47.12.48.12:4401 Unknown Yes i2050 65-01
65-01
(192.168.0.101:5000) (192.168.0.101:5200)

Total sets=3
```

The public and private IP address and ports are provided for the signaling and media. If the private signaling port is not available, it is not printed. The NAT type is indicated (as detected by the Port Mapping Discovery with the second ES). The output indicates whether RTCP signaling is supported; if “No”, then features that depend on RTCP may not work correctly. The feature that uses RTCP is the QoS Monitoring feature, otherwise known as Proactive Voice Quality Management (PVQM). For RTCP feature to work, the NAT device must assign the public RTCP port for a telephone to equal RTP+1 (that is, RTCP Public Port = RTP Public Port +1). In many cases, the NAT device assigns public ports sequentially, so the RTCP port is RTP+1 (for example 10002 for RTP and 10003 for RTCP). Thus, the PVQM feature works. If the NAT device does not assign the Public Ports sequentially, the PVQM feature cannot work.

> ***Note:*** The NAT device can assign the public port number for RTCP to be the public RTP port number +1, but then the mapping cannot be maintained because the wrong firmware is present (alarm ITG3057) or the device is an undetected symmetric NAT (marked as Unknown NAT). The availability of RTCP signaling still shows as "Yes" in these cases."

Enter the isetNATShow command at the CLI of any card in a VGMC node along with the TN or public IP address of a particular IP Phone to display the telephone information, along with the identification of the card with which the IP Phone is registered. This data is useful when you need to identify which card to enable a message monitor on, or connect a sniffer to, when you debug a particular IP Phone problem, as shown in the following example.

```
VGMC>isetNatShow “47.11.222.45”
value = 0 = 0x0
VGMC>
signaling Media Public IP Addr:Port Public IP Addr:Port
(Private IP Addr:Port) (Private IP Addr:Port) NAT Type RTCP
Type Set-TN Reg-TN
---------------------- ------------------------
--------- ---- ------ ------------ ------------
->Found on Card TN 002-00 , ELAN IP 47.11.214.52, TLAN
IP 47.100.1.2:  47.11.222.45:6200 <No speech possible>
Symmetric No i2004 64-03 64-09 (192.168.0.100:5000)
VGMC>
```

This command also handles the same public IP address that appears for multiple phones (one NAT device has multiple phones on it, as shown in the following example.

```
VGMC> isetNATShow "47.11.213.114"
signaling Media Public IP Addr:Port Public IP Addr:Port
(Private IP Addr:Port) (Private IP Addr:Port) NAT Type RTCP
Type Set-TN Regd-TN
---------------------------- ----------------------------
- --------- ---- ------------ ------------- ------------
->Found on Card TN 002-00 , ELAN IP 47.11.214.52, TLAN
IP 47.100.1.2:  47.11.213.114:1224 47.11.213.114:1698
Cone Y i2004 061-01 061-01 (192.168.0.102:5000)
(192.168.0.102:5200) ->Found on Card TN 009-00 , ELAN IP
47.11.217.21, TLAN IP 47.11.215.185:  47.11.213.114:1225
47.11.213.114:1700 Cone Y i2004 061-00 061-00
(192.168.0.101:5000) (192.168.0.101:5200)
```

## isetReset

Syntax:

**`isetReset "phoneIPAddr" or "l s c u"`**

Reset a single IP Phone. Specify the IP Phone by either its IP address (phoneIPAddr) or VTN (l s c u). The IP address is the public signaling IP address of the telephone.

The following example shows both uses of the command.

```
VGMC> isetReset "192.168.1.141"
value = 0 = 0x0
VGMC>
FEB 16 15:32:40 tShell:  Info Reset i2004 set with IP
192.168.1.141
FEB 16 15:32:58 tCSV: Info 192.168.1.141 Connecting to node
1, TN: 61.1, 0x6005
FEB 16 15:33:00 tRDP: Warning 192.168.1.141 Connection
restarted, cid = 0x33cc350
FEB 16 15:33:00 tVTM: Notice 192.168.1.141 Unregistered,
terminal = 0x3a876fc, device = 0x33cb904
FEB 16 15:33:00 tSET: Info Terminal offline 192.168.1.141
TN 0x0000
FEB 16 15:33:01 tVTM: Info ITS5008 Terminal connection
status:  192.168.1.141 ok(20)
FEB 16 15:33:01 tSET: Info 192.168.1.141 TN 61-01
Registered with M1

VGMC> isetReset 61-01
Invalid input:  isetReset "l s c u" or isetReset "IP"
Example:  isetReset "61 0 0 1" or isetReset "192.168.1.2"
value = 59 = 0x3b = ';'
VGMC> isetReset "61 1"
```

```
value = 0 = 0x0
VGMC>
FEB 16 15:35:10 tShell:  Info Reset i2004 set with TN
0x6005:  61 1 ...
FEB 16 15:35:32 tSET: Info 192.168.1.141 TN 61-01
Registered with M1
```

If the IP address identifies multiple IP Phones (for example, multiple IP Phones are on a NAT that shares the same public IP address), then an error message prints. This message indicates more than one IP Phone with the IP address and recommends using the isetReset “TN” command.

The following example demonstrates the isetReset command output for cases where there are two IP Phones with the same public IP address.

```
-> isetReset "47.11.215.183"

WARNING: There are 2 IP Phones that use the public IP address
of 47.11.215.183
Please reset the Internet Telephone using the TN: isetReset
“TN”.
```

## isetResetAll

Syntax:

**`isetResetAll`**

Reset all IP Phones registered with theTPS.

```
VGMC> isetResetAll
value = 0 = 0x0
VGMC>
FEB 16 15:38:25 tShell:  Info Reset all registered i2004 set
```

## isetScpwQuery

Syntax:

**`isetScpwQuery “IPAddr” or “TN”`**

Print the status of the station control password (SCPW). This password is used for the VO and PD VGMC features. Enter the IP Phone public IP address or TN as the command parameter.

```
-> isetScpwQuery "47.11.215.136"
value = 0 = 0x0
-> 21/01/04 14:12:05 LOG0006 SET: SCPW Query Status (tn
61-1):  Defined

-> isetScpwQuery "61 1"
```

```
value = 0 = 0x0
-> 21/01/04 14:12:26 LOG0006 SET: SCPW Query Status (tn
61-1):  Defined
```

## isetScpwVerify

Syntax:
**`isetScpwVerify "IPAddr" or "TN", "password"`**

Verify the entered SCPW matches what is currently configured. Enter the IP Phone public IP address or TN, and the existing SCPW as the command parameters.

```
-> isetScpwVerify "147.11.215.136","1234"
value = 0 = 0x0
-> 22/01/04 15:21:08 LOG0006 SET: SCPW Verify Status (tn
61-1):  OK
```

## isetScpwModify

Syntax:
**`isetScpwModify "IPAddr" or "TN", "password"`**

Modify the SCPW from the VGMC application rather than from the CS 1000. Enter the IP Phone public IP address or TN and the new SCPW as the command parameters.

```
->isetScpwModify "147.11.215.136","6100"
value = 0 = 0x0
-> 22/01/04 15:23:45 LOG0006 SET: SCPW Modify Status (tn
61-1):  OK
```

## isetShow, isetShowByTN, isetShowByIP

Syntax:
**`isetShow`**
**`isetShowByTN`**
**`isetShowByIP`**

These commands list every IP Phone registered with the TPS. For each telephone, the IP Phone status, type of registration, time up (registered to TPS), virtual TN, MAC address, type of IP Phone, firmware and UNIStim signaling versions, and the RUDP ports for the IP Phone and TPS are shown. The isetShowByTN and isetShowByIP commands display the output list in order by TN and IP address, respectively, while the isetShow command lists the IP Phones in order of registration.

Enter the isetShow command at the CLI of any card in a VGMC node along with the TN or IP address of a particular IP Phone to print this IP Phone information, along with the identification of the card with which the

IP Phone is registered. This is useful when you need to identify which card to enable a message monitor on, or to connect a sniffer to, when you debug a particular IP Phone problem. You can also achieve this by using the IDU command from LD 32 on the Call Server, if you know the TN.

If you enter a particular IP address or TN but no card responds with the information, an error message prints after the "wait for card responses" timeout expires.

The regType field indicates the type of registration:

- Regular: default registration; non-branch office Internet terminal registered with its "home" TPS (the TPS defined by the telephone S1/S2 configuration).
- Virtual: virtual office registration; Internet terminal "home" TPS is another system but its user has invoked the virtual office feature causing the telephone to register on this TPS.
- Branch: branch office Internet terminal registered at the main office. The main office TPS displays this type of registration.
- Local: branch office IP Phone registered at the branch office. Internet terminal is in local mode; the branch office TPS displays this type of registration.

You can check the Up Time to see if an IP Phone or group of IP Phones is resetting (the Up Time is less than the other telephones). This may indicate a network problem if a group of phones in a specific subnet reboot at about the same time.

Ensure that the firmware version is the same on all IP Phones. Compare the version shown to the output of the umsPolicyShow command.

An IP Phone MAC address is in the middle of the Hardware ID, separated by dashes from the remainder of the Hardware ID.

A single letter indicates the NAT type detected for an IP Phone. The NAT column value is valid when the State column indicates the telephone is registered to the CS 1000. Possible values are:
C = Cone NAT
S = Symmetric NAT
U = Unknown: behind a NAT of unknown type (response received from only ES1)
P - Pending: waiting on response from the telephone or the telephone never received a response from ES1
" " (space) = the IP Phone is not behind any NAT (normal case)

IP Phones with NAT type S are unregistered from the CS 1000.

```
VGMC> isetShow
IP Address Type RegType State Up Time TN HWID FWVsn
UNIStimVsn SrcPort DstPort
------------------ ------- ------- -----------
------------- ------------ ------------------- -------
---------- ------ -------
10.1.1.5 i2004 Regular online 0 02:03:17 061-01
18006038764lf36600 0602B39 2.5 5100 5000
10.1.1.4 i2004 Regular online 0 00:01:00 061-00
18006038764lc16600 0602B39 2.5 5100 5000

Total sets = 2
value = 0 = 0x0

VGMC> isetShow "47.11.213.114"
IP Address Type RegType State Up Time TN HWID FWVsn
UNIStimVsn SrcPort DstPort
------------------ ------- ------- -----------
------------- ------------ ------------------- -------
---------- ------ -------
10.1.1.5 i2004 Regular online 0 02:03:17 061-01
18006038764lf36600 0602B39 2.5 5100 5000
10.1.1.4 i2004 Regular online 0 00:01:00 061-00
18006038764lc16600 0602B39 2.5 5100 5000

> Found on Card TN 010-00 , ELAN IP 47.11.216.48, TLAN IP
47.11.215.53
47.11.213.114 i2001 Regular online 0 21:37:14 061-04 061-04
1800-oae402e283-6602 0604A307 5100 5000

> Found on Card TN 009-00 , ELAN IP 47.11.217.21, TLAN IP
47.11.215.185
47.11.113.114 C i2004 Regular online 0 04:20:34 061-00
061-00 1800-6038b689a9-6600 0602B59 5100 6123

VGMC> isetShow "61 1"
value = 0 = 0x0
VGMC-
IP Address Type RegType State Up Time TN HWID FWVsn
UNIStimVsn SrcPort DstPort
------------------ ------- ------- -----------
------------- ------------ ------------------- -------
---------- ------ -------
10.1.1.5 i2004 Regular online 0 02:03:24 061-01
18006038764lf36600 0602B39 2.5 5100 5000
VGMC> isetShow "61 23"
value = 0 = 0x0
```

```
VGMC>
**** Node Information Request time out Alert ****
Request #1 Type:  get set info
Variables[1]:  TN 061-23
```

## itgA07TraceHelp

Syntax:

**`itgA07TraceHelp`**

This command displays the CLI commands and parameters.

## itgA07TraceSettings

Syntax:

**`itgA07TraceSettings`**

Print the current trace settings.

## itgA07TraceSetOutput

Syntax:

**`itgA07TraceSetOutput trace_output, "file_name"`**

This command assigns the output destination for the trace tool. The parameter trace_output is an integer value specifying the trace output destination:
1 = TTY
2 = SYSLOG
3 = File
4 = File and TTY

The file_pathname is a string encapsulated in quotation marks that specifies the file to output to if trace_output = 3 or 4. You can enter the entire path name, or only the file name (in which case the default path is C:/file_name).

```
-> itgA07TraceSetOutput 1, "trace.txt"
value = 0 = 0x0
```

## itgA07TraceOff

Syntax:

**`itgA07TraceOff chNum`**

This command turns off the trace for the specific channel number chNum.

## itgA07TraceAllOff

Syntax:

**`itgA07TraceAllOff`**

This command turns off the trace for all channels.

### itgA07TraceOn

Syntax:

**`itgA07TraceOn chNum`**

This command turns on the trace for one channel.

```
-> itgA07TraceOn 10 value = 0 = 0x0
```

### itgAlarmTest

Syntax:

**`itgAlarmTest alarm`**

Create and send a dummy SNMP alarm for one or more ITGnnnn alarms. To send dummy SNMP alarm messages for all ITS alarms, assign **`alarm`** a value less than 0. To send a dummy SNMP alarm message for the specified alarm, assign **`alarm`**
a value from 0 to 35.

```
VGMC> itgAlarmTest 35
value = 0 = 0x0 VGMC>
FEB 16 15:58:11 tShell:  Notice ITG6035 Encountered an
unexepected open DSP chann el, closed it:  11 (202)
VGMC>
```

### itgCardShow

Syntax:

**`itgCardShow`**

Display the card role, TLAN IP addresses, system information and location, status, and up time.

The following table describes the data parameters printed by this command.

**Table 42**
**Data output**

| Parameter | Description |
|---|---|
| Leader IP | An IP address on the TLAN interface.<br><br>If Leader IP is 0.0.0.0, either the BOOTP.TAB file is missing or does not parse properly. Ensure that the file is in the CONFIG directory. Look for alarms or error messages on the TTY console. |

**Table 42**
**Data output (cont'd.)**

| Parameter | Description |
|---|---|
| Card IP | An IP address on the TLAN interface.<br><br>If card IP is 0.0.0.0, the card cannot locate or communicate with the BOOTP server or Leader card. Ensure that the Leader card has the Follower entry with the correct management MAC address in the BOOTP.TAB file. If correct, check the TLAN and ELAN cable connections. Next, check for errors or alarms on the TTY console and on MAT. |
| ELAN (lnIsa) stat<br><br>ELAN (ixpMac1) stat | Interface speed and carrier status of the ELAN interface. |
| TLAN (lnPci) stat<br><br>TLAN (ixpMac0) | Interface speed and carrier status of the TLAN interface.<br><br>If the carrier is not OK for an interface, check the cable and switch connected to that interface. |
| Card State | ENBL = The card is configured and operational<br><br>DSBL = The card is configured but out of service. Enable from Call Server LD 32.<br><br>UNEQ = The card is not configured in the Call Server. |
| Card TN | The VGMC TN prints.<br><br>This field is omitted for the Signaling Server. |

The following example shows output on an ITG-P card.

```
VGMC> itgCardShow
Index :  1
Type :  EXUT
Role :  Leader
Node :  888
Leader IP : 47.11.254.2
Card IP : 47.11.254.3
Card TN : Slot 20
Card State :  ENBL
Uptime :  0 days, 7 hours, 59 mins, 20 secs (28760 secs)
Codecs :  G711Ulaw(default), G711Alaw, G729AB, G723_5,
G711CC, T38FAX
ELAN (lnIsa) stat :  10 Mbps, Half duplex (Carrier OK)
TLAN (lnPci) stat :  100 Mbps, Full duplex (Carrier OK)
value = 1 = 0x1
```

Example of the data printed on the SMC:

```
-> itgCardShow
Index :  2
Type :  EXUT
Role :  Leader
Node :  123
Leader IP : 47.11.215.216
Card IP : 47.11.215.214
Card TN : Slot 3
Card State :  ENBL
Uptime :  0 days, 0 hours, 9 mins, 34 secs (574 secs)
Codecs :  G711Ulaw(default), G711Alaw, G729AB, G711CC,
T38FAX ELAN (ixpMac1) stat:  10 Mbps, Half duplex (Carrier
OK)
TLAN (ixpMac0) stat:  100 Mbps, Full duplex (Carrier OK)
```

The following example shows output printed on the Signaling Server.

```
-> itgCardShow
Index :  1
Role :  Leader
Node :  777
Leader IP : 47.11.249.105
Card IP : 47.11.249.106
Uptime :  0 days, 0 hours, 9 mins, 10 secs (550 secs)
Codecs :  G711Ulaw(default), G711Alaw, G729A, G729AB,
G723_5, G711CC, T38FAX
value = 1 = 0x1
```

## itgChanStateShow

Syntax:
**`itgChanStateShow`**

List the state (busy, idle, disabled, or unequipped) of all channels on a VGMC. You can use this command to check if there are active calls on the card, or to see which channel is being used by the call. A channel is marked busy when a call is active on it.

```
VGMC> itgChanStateShow
Channel 0 :  Busy
Channel 1 :  Disabled
Channel 2 :  Busy
Channel 3 :  Unequipped
Channel 4 :  Idle
Channel 5 :  Idle
...  (one line per card channel)
Channel 22 :  Idle
Channel 23 :  Idle

value = 1 = 0x1
```

### itgMemShow

Syntax:

**`itgMemShow`**

Print general information about the memory available on the VGMC. On the Signaling Server, this command displays the same information as the memShow command. This command displays free memory and the blocks it is composed of, memory in use, and cumulative allocated memory.

```
VGMC> itgMemShow
status bytes blocks avg block max block
------ --------- -------- ---------- ----------
current
free 49448328 77 642186 49292668
alloc 12667292 2395 5289 -
cumulative
alloc 40264460 311809 129 -
value = 0 = 0x0

VGMC>
```

### itgMsgQShow

Syntax:

**`itgMsgQShow`**

Print general information about the various message queues used by the VGMC application:

- queue ID
- task in the queue
- number of messages sent over the queue
- number of lost messages

The following example shows command output on the ITG-P card.

```
-> itgMsgQShow
|QID |Nbr of Use|Pipe FD|Queue Name |Pipe Name |Desc(h) |Nbr
Msgs |High Mark |Lost Msgs |
+-----+----------+-------+------------+---------------+
----------+---------+---------+----------+
|6 |1 |23 |/msgq.6 |/pipe/srv.6 |0x1870afc |0 |1 |0 |
|30 |1 |-1 |/msgq.30 |/pipe/srv.30 |0x17daebc |0 |1 |0 |
|31 |1 |-1 |/msgq.31 |/pipe/srv.31 |0x18703c8 |0 |2 |0 |
|32 |1 |-1 |/msgq.32 |/pipe/srv.32 |0x18702c0 |0 |35 |0 |
|33 |1 |-1 |/msgq.33 |/pipe/srv.33 |0xfb3c4c |0 |3 |0 |
|35 |1 |-1 |/msgq.35 |/pipe/srv.35 |0x1856ad8 |0 |2 |0 |
|38 |1 |46 |/msgq.38 |/pipe/srv.38 |0x17c4778 |0 |3 |0 |
|39 |1 |38 |/msgq.39 |/pipe/srv.39 |0x1831958 |0 |2 |0 |
|40 |1 |-1 |/msgq.40 |/pipe/srv.40 |0xfb79d8 |0 |1 |0 |
|41 |1 |-1 |/msgq.41 |/pipe/srv.41 |0xfb7e5c |0 |2 |0 |
|42 |1 |-1 |/msgq.42 |/pipe/srv.42 |0x17c5ca0 |0 |2 |0 |
|43 |1 |-1 |/msgq.43 |/pipe/srv.43 |0x183eb44 |0 |30 |0 |
|57 |1 |-1 |/msgq.57 |/pipe/srv.57 |0xfb8ec0 |0 |0 |0 |

value = 2079 = 0x81f
->

OCT 24 13:37:51 tShell:  Info
OCT 24 13:37:51 tShell:  Info +----------+---------+-----
-----+
OCT 24 13:37:51 tShell:  Info |Category |Cat ID(h) |MsgQ
ID(d)|
OCT 24 13:37:51 tShell:  Info +----------+---------+-----
-----+
OCT 24 13:37:51 tShell:  Info |LOG |0xc00 |3 |
OCT 24 13:37:51 tShell:  Info |RUDP |0x1800 |6 |
OCT 24 13:37:51 tShell:  Info |MYCAT |0x2000 |8 |
OCT 24 13:37:51 tShell:  Info |VTM |0x7800 |30 |
OCT 24 13:37:51 tShell:  Info |VTI |0x7c00 |31 |
OCT 23 13:37:51 tShell:  Info |DSET |0x8000 |32 |
OCT 23 13:37:51 tShell:  Info |VGW |0x8400 |33 |
OCT 23 13:37:51 tShell:  Info |MAM |0x8c00 |35 |
OCT 23 13:37:51 tShell:  Info |TPS |0x9800 |38 |
OCT 23 13:37:51 tShell:  Info |ELC |0x9c00 |39 |
OCT 23 13:37:51 tShell:  Info |UMSClientS|0xa000 |40 |
OCT 23 13:37:51 tShell:  Info |UMSServerS|0xa400 |41 |
OCT 23 13:37:51 tShell:  Info |CSV |0xa800 |42 |
OCT 23 13:37:51 tShell:  Info |OMM |0xac00 |43 |
OCT 23 13:37:51 tShell:  Info |TPSAR |0xe000 |56 |
OCT 23 13:37:51 tShell:  Info |UMSFwBk |0xe400 |57 |
OCT 23 13:37:51 tShell:  Info +----------+---------+-----
-----+
->
```

Example SMC output:

```
-> itgMsgQShow
|QID |Nbr of Use|Pipe FD|Queue Name |Pipe Name |Desc(h) |Nbr
Msgs |High Mark |Lost Msgs |
+-----+---------+------+----------+-------------+
----------+---------+---------+----------+
|6 |1 |20 |/msgq.6 |/pipe/srv.6 |0x2c03f3c |0 |2 |0 |
|30 |1 |-1 |/msgq.30 |/pipe/srv.30 |0x2b68c78 |0 |1 |0 |
|31 |1 |-1 |/msgq.31 |/pipe/srv.31 |0x2b62248 |0 |1 |0 |
|32 |1 |-1 |/msgq.32 |/pipe/srv.32 |0x2b58408 |0 |1 |0 |
|33 |1 |-1 |/msgq.33 |/pipe/srv.33 |0x23166e0 |0 |3 |0 |
|35 |1 |-1 |/msgq.35 |/pipe/srv.35 |0x2be8e84 |0 |2 |0 |
|38 |1 |44 |/msgq.38 |/pipe/srv.38 |0x2b4b2cc |0 |8 |0 |
|39 |1 |34 |/msgq.39 |/pipe/srv.39 |0x2bbff54 |0 |1 |0 |
|40 |1 |-1 |/msgq.40 |/pipe/srv.40 |0x231e8c4 |0 |1 |0 |
|41 |1 |-1 |/msgq.41 |/pipe/srv.41 |0x2343e94 |0 |4 |0 |
|42 |1 |-1 |/msgq.42 |/pipe/srv.42 |0x2b53a88 |0 |5 |0 |
|43 |1 |-1 |/msgq.43 |/pipe/srv.43 |0x2bcd160 |0 |2 |0 |
|56 |1 |-1 |/msgq.56 |/pipe/srv.56 |0x2b4c7e0 |0 |0 |0 |
|57 |1 |-1 |/msgq.57 |/pipe/srv.57 |0x23452f8 |0 |0 |0 |
value = 100 = 0x64 = 'd'
->
OCT 24 13:49:34 tShell:  Info
OCT 24 13:49:34 tShell:  Info +----------+---------+-----
-----+
OCT 24 13:49:34 tShell:  Info |Category |Cat ID(h) |MsgQ
ID(d)|
OCT 24 13:49:34 tShell:  Info +----------+---------+-----
-----+
OCT 24 13:49:34 tShell:  Info |LOG |0xc00 |3 |
OCT 24 13:49:34 tShell:  Info |RUDP |0x1800 |6 |
OCT 24 13:49:34 tShell:  Info |MYCAT |0x2000 |8 |
OCT 24 13:49:34 tShell:  Info |VTM |0x7800 |30 |
OCT 24 13:49:34 tShell:  Info |VTI |0x7c00 |31 |
OCT 24 13:49:34 tShell:  Info |DSET |0x8000 |32 |
OCT 24 13:49:34 tShell:  Info |VGW |0x8400 |33 |
OCT 24 13:49:34 tShell:  Info |MAM |0x8c00 |35 |
OCT 24 13:49:34 tShell:  Info |TPS |0x9800 |38 |
OCT 24 13:49:34 tShell:  Info |ELC |0x9c00 |39 |
OCT 24 13:49:34 tShell:  Info |UMSClientS|0xa000 |40 |
OCT 24 13:49:34 tShell:  Info |UMSServerS|0xa400 |41 |
OCT 24 13:49:34 tShell:  Info |CSV |0xa800 |42 |
OCT 24 13:49:34 tShell:  Info |OMM |0xac00 |43 |
OCT 24 13:49:34 tShell:  Info |TPSAR |0xe000 |56 |
OCT 24 13:49:34 tShell:  Info |UMSFwBk |0xe400 |57 |
OCT 24 13:49:34 tShell:  Info +----------+----------+--
--------+
```

The following example shows command output on the ITG-P card.

```
-> itgMsgQShow
0xaada058 (tShell):
0xaada058 (tShell):  +----------+---------+----------+
0xaada058 (tShell):  |Category |Cat ID(h) |MsgQ ID(d)|
0xaada058 (tShell):  +----------+---------+----------+
0xaada058 (tShell):  |LOG |0xc00 |3 |
0xaada058 (tShell):  |RUDP |0x1800 |6 |
0xaada058 (tShell):  |MYCAT |0x2000 |8 |
0xaada058 (tShell):  |MAM |0x8c00 |35 |
0xaada058 (tShell):  |ELC |0x9c00 |39 |
0xaada058 (tShell):  |OMM |0xac00 |43 |
0xaada058 (tShell):  |GKNPM |0xc400 |49 |
0xaada058 (tShell):  |GKHTTP |0xc800 |50 |
0xaada058 (tShell):  |GKDBM |0xcc00 |51 |
0xaada058 (tShell):  |GKOMM |0xd000 |52 |
0xaada058 (tShell):  |HTTP |0xd400 |0 |
0xaada058 (tShell):  |HTTP |0xd800 |0 |
0xaada058 (tShell):  +----------+---------+----------+

|QID |Nbr of Use|Pipe FD|Queue Name |Pipe Name |Desc(h) |Nbr
Msgs |High Mark |Lost Msgs |
+-----+----------+-------+------------+--------------+
----------+---------+---------+----------+
|6 |1 |20 |/msgq.6 |/pipe/srv.6 |0xba10ef4 |0 |10 |0 |
|35 |1 |-1 |/msgq.35 |/pipe/srv.35 |0xb9f51a0 |0 |1 |0 |
|39 |1 |28 |/msgq.39 |/pipe/srv.39 |0xb9df6e4 |0 |1 |0 |
|43 |1 |-1 |/msgq.43 |/pipe/srv.43 |0xb9f04a4 |0 |1 |0 |
|49 |1 |29 |/msgq.49 |/pipe/srv.49 |0xb89528c |0 |1 |0 |
|51 |1 |-1 |/msgq.51 |/pipe/srv.51 |0xb9c03b0 |0 |40 |0 |
|52 |1 |-1 |/msgq.52 |/pipe/srv.52 |0xb9a2898 |0 |1 |0 |

value = 100 = 0x64 = 'd'
```

## itgPLThreshold

Syntax:

**`itgPLThreshold threshold`**

Assign the threshold for packet loss which, if exceeded during a call, generates an alarm at the end of the call. The parameter threshold is a value from 1 to 1000 in units of a tenth of a percent. The default threshold is 10 (that is, 1 percent).

When a call through the VGMC gateway is released, the application compares the number of packets lost with the total number of packets. The Transmit (TX) and Receive (RX) streams (to and from the terminal, respectively) are checked individually. If the percentage exceeds the configured threshold value, an alarm occurs.

The alarm can be reported from two tasks: tRTP logs packet loss for data through the gateway, and tVTM logs packet loss reported by the IP Phone. The alarm shows the channel number, the percentage of packets lost, the direction of the packet stream (RX is IP Phone to VGMC), the IP Phone IP address and the alarm cause field (26 is alarmCauseCongestion).

```
VGMC> itgPLThreshold 50
value = 0 = 0x0
VGMC>
SEP 22 11:17:48 tRTP: Warning ITG4028 Voice packet loss:  0
6.4% rx 47.147.75.81 (26)
SEP 22 11:17:50 tVTM: Warning ITG4028 Voice packet loss:  0
6.6% tx 47.147.75.81 (26)
```

## itgShell

Syntax:
**`itgShell`**

Returns to the VGMC command line interface prompt (VGMC>). This command applies to VGMC only and is not supported on MC32S.

```
-> itgShell
VGMC>
```

## itsAlarmTest

Syntax:
**`itsAlarmTest alarm`**

Creates and sends a dummy SNMP alarm for one or more ITSnnnn alarms.

To send dummy SNMP alarm messages for all ITS alarms, assign **`alarm`** a value less than 0. To send a dummy SNMP alarm message for the specified alarm, assign **`alarm`** a value from 0 to 10.

```
VGMC> itsAlarmTest 2
value = 0 = 0x0
VGMC>
FEB 16 15:51:29 tShell:  Notice ITS6002 Connect service
activation:  Call server i s upside down (202)
VGMC>
```

### lastResetReason

Syntax:
**`lastResetReason`**

Display the reason for the last reset of the VGMC.

The following table describes the data parameters printed by this command.

**Table 43**
**Data output**

| Parameter | Description |
|---|---|
| Reboot command issued | Output after card reset using the CLI command cardReboot. |
| Watchdog Timer Expired | Output after card reset due to watchdog timer expiration. |
| Manual reset | Output after card reset after faceplate reset button is pressed or after a power cycle to the card. |
| Unknown | Output after card reset because the card firmware does not support the reset reason, or the reset reason code is corrupt. |

```
VGMC> lastResetReason
Last Reset Reason:  Unknown
value = 1 = 0x1
VGMC>
```

### ll

Syntax:
**`ll`**

List the contents of the current directory with timestamp and size information. In addition to comparing file timestamps, this command is useful to check for zero-length files, which may indicate disk corruption.

```
-> ll
size date time name
-------- ------ ------ --------
```

```
512 JAN-01-1996 12:06:34 .  <DIR>
512 JAN-01-1996 12:06:34 ..  <DIR>
1001 JAN-01-1996 12:15:14 CONFIG.INI
958 JUL-26-2000 15:07:52 CONFIG.BAK
225 JAN-01-1996 16:08:54 BOOTP.TAB
225 JAN-01-1996 16:12:50 BOOTP.BAK
197 JAN-01-1996 14:25:38 UMS.INI
value = 0 = 0x0
```

### lnIsaBroadcastShow

Syntax:

**`lnIsaBroadcastShow`**

Display the number of broadcast IP packets received on the lnIsa (ELAN) interface every second for the past 60 seconds. This command is available only on the ITG-P. The total number of broadcast messages received on the interface since the card was booted also appears.

```
-> lnIsaBroadcastShow
Broadcasts per second, last 60 seconds:  lnIsa0

0:  11 12 8 3 6 12
6:  9 9 10 5 11 21
12:  19 11 6 4 4 14
18:  21 12 2 9 5 20
24:  12 13 19 13 2 5
30:  15 24 13 29 19 28
36:  24 20 3 2 1 2
42:  3 2 9 2 9 18
48:  13 3 11 5 5 19
54:  14 6 16 11 9 13
60:  4
Total broadcasts since startup = 938427
value = 41 = 0x29 = ')'
```

### lnIsaBroadcastThreshold = threshold

Enable this variable to control the threshold (in packets per second) at which a message prints when you configure lnIsaShowLostBroadcast. The default threshold is 40. This variable applies to the ITG-P only.

```
-> lnIsaBroadcastThreshold = 20
_lnIsaBroadcastThreshold = 0x34aeb0:  value = 20 = 0x14
```

### lnIsaShowLostBroadcast = 1/0

This variable applies to the ITG-P only. Enable this variable to print a message whenever the broadcast threshold (set through the lnIsaBroadcastThreshold variable) is exceeded on the lnIsa (ELAN) interface. This is useful to debug broadcast storms. The default is off (0); the state is not saved and returns to off if the card is reset or reboots.

The following example occurs when the threshold is 0, which is not a normal setup.

```
-> lnIsaShowLostBroadcast = 1
_lnIsaShowLostBroadcast = 0x34aec8:  value = 1 = 0x1

FEB 12 12:23:27 tLogTask:  Info Time 332140 lnIsa:  1
broadcasts in last second
```

### lnPciBroadcastShow

Syntax:
**`lnPciBroadcastShow`**

Display the number of broadcast IP packets received on the lnPci (TLAN) interface every second for the past 60 seconds. This command is available only on the ITG-P. The total number of broadcast messages received on the interface since the card was booted also appears. This command is not available on the Signaling Server.

```
-> lnPciBroadcastShow
Broadcasts per second, last 60 seconds:  lnPci1
```

```
0: 0 0 0 0 0 0
6: 0 0 0 0 0 0
12: 0 0 0 0 0 0
18: 0 0 0 0 0 0
24: 0 0 0 0 0 0
30: 0 0 0 0 0 0
36: 0 0 0 0 0 0
42: 0 0 0 0 0 0
48: 0 0 0 0 0 0
54: 0 0 0 0 0 0
60: 0
Total broadcasts since startup = 769
value = 37 = 0x25 = ’%’
```

### lnPciBroadcastThreshold = threshold

This variable applies only to the ITG-P. Enable this variable to control the threshold (in packets per second) at which a message prints when the lnPciShowLostBroadcast is set. The default threshold is 40.

```
-> lnPciBroadcastThreshold = 200
_lnPciBroadcastThreshold = 0x34b008:  value = 200 = 0xc8
```

## lnPciShowLostBroadcast = 1/0

This variable applies only to the ITG-P. Enable this variable to print a message whenever the broadcast threshold (set through the lnPciBroadcastThreshold variable) is exceeded on the lnPci (TLAN) interface. This is useful to debug broadcast storms. The default is off (0); the state is not saved and returns to off if the card is reset or reboots.

The following example occurs when the threshold is 0, which is not a normal setup.

```
-> lnPciShowLostBroadcast = 1
_lnPciShowLostBroadcast = 0x34b020:  value = 1 = 0x1

FEB 12 12:23:27 tLogTask:  Info Time 332140 lnPci:  1
broadcasts in last second
```

## loadBalance

Syntax:
**`loadBalance`**

Redistribute the registration of IP Phones across the TPSs in the node. If phones register to VGMCs while a Signaling Server is out of service, this command is the fastest way to gracefully migrate the phones back to the node Signaling Servers. You can run this command from the shell of any card in the node (Signaling Server, SMC, or ITG-P).

The following example shows the output on the Signaling Server, where three phones are moved from a VGMC to the Signaling Server.

```
oam> isetShow

Set Information
---------------
No sets registered

oam> loadBalance oam>
07/11/03 13:40:24 LOG0006 shell:  Line TPS is attempting to
balance the registration load of sets between this card and
the rest of the node components
07/11/03 13:40:29 LOG0006 TPS: Average number of sets
registered to a signaling server should be:  3.  Average
number of sets registered to a VGMC card should be:  0
07/11/03 13:40:29 LOG0006 SET: Load balance message
received from TPS
```

```
07/11/03 13:41:17 LOG0004 CSV: ITS4011 47.11.215.135
connecting to node 3556, TN: 61-00, 0x6004 [Last Reset
Reason Not Supported, code:  127] (201)
07/11/03 13:41:22 LOG0006 VTM: ITS5008 Terminal connection
status:  47.11.215.135 ok (20)
07/11/03 13:41:22 LOG0006 SET: 47.11.215.135 TN 61-00
Registered with CS
07/11/03 13:41:39 LOG0004 CSV: ITS4011 47.11.215.230
connecting to node 3556, TN: 61-05, 0x6045 [Last Reset
Reason Not Supported, code:  127] (201)
07/11/03 13:41:40 LOG0004 CSV: ITS4011 47.11.215.136
connecting to node 3556, TN: 61-01, 0x6005 [Last Reset
Reason Not Supported, code:  127] (201)
07/11/03 13:41:44 LOG0006 VTM: ITS5008 Terminal connection
status:  47.11.215.230 ok (20)
07/11/03 13:41:44 LOG0006 SET: 47.11.215.230 TN 61-05
Registered with CS
07/11/03 13:41:45 LOG0006 VTM: ITS5008 Terminal connection
status:  47.11.215.136 ok (20)
07/11/03 13:41:45 LOG0006 SET: 47.11.215.136 TN 61-01
Registered with CS
```

The following messages print on VGMC at the same time.

```
NOV 07 13:40:28 SET: Info Load balance message received from
TPS
NOV 07 13:40:28 SET: Info 3 number of idle sets need to do
service switch
NOV 07 13:40:28 SET: Info loadBalance has been completed
NOV 07 13:40:44 RDP: Error ITS2008 Terminal connection
status:  47.11.215.135 lost (20)
NOV 07 13:40:44 VTM: Notice 47.11.215.135 Unregistered,
terminal = 0x2e20a68, device = 0x2e20b70
NOV 07 13:40:44 SET: Info 47.11.215.135 TN 61-00 Terminal
offline
NOV 07 13:40:45 RDP: Error ITS2008 Terminal connection
status:  47.11.215.136 lost (20)
NOV 07 13:40:45 VTM: Notice 47.11.215.136 Unregistered,
terminal = 0x2301ca4, device = 0x22ff3c4
NOV 07 13:40:45 SET: Info 47.11.215.136 TN 61-01 Terminal
offline
NOV 07 13:40:46 RDP: Error ITS2008 Terminal connection
status:  47.11.215.230 lost (20)
NOV 07 13:40:46 VTM: Notice 47.11.215.230 Unregistered,
terminal = 0x2300fb8, device = 0x2e21500
NOV 07 13:40:46 SET: Info 47.11.215.230 TN 61-05 Terminal
offline ->
```

The following example shows command output on the ITG-P card, where two phones are moved from the card to the Signaling Server.

```
VGMC> loadBalance
value = 0 = 0x0
VGMC>
NOV 07 13:53:12 tShell:  Info Line TPS is attempting to
balance the registration load of sets between this card and
the rest of the node components
NOV 07 13:53:17 TPS: Info Average number of sets registered
to a signaling server should be:  3.  Average number of sets
registered to a VGMC card should be:  0
NOV 07 13:53:17 SET: Info Load balance message received from
TPS NOV 07 13:53:17 SET: Info 2 number of idle sets need to do
service switch
NOV 07 13:53:17 SET: Info loadBalance has been completed
VGMC> VGMC>
NOV 07 13:53:32 RDP: Error ITS2008 Terminal connection
status:  47.11.215.230 lost (20)
NOV 07 13:53:32 VTM: Notice 47.11.215.230 Unregistered,
terminal = 0xf8da98, device = 0xf8b714
NOV 07 13:53:32 SET: Info 47.11.215.230 TN 61-05 Terminal
offline
NOV 07 13:53:33 RDP: Error ITS2008 Terminal connection
status:  47.11.215.136 lost (20)
NOV 07 13:53:33 VTM: Notice 47.11.215.136 Unregistered,
terminal = 0xf8d184, device = 0xf8ce94
NOV 07 13:53:33 SET: Info 47.11.215.136 TN 61-01 Terminal
offline VGMC>
```

The following messages print on the Signaling Server at the same time.

```
oam> 07/11/03 13:53:18 LOG0006 SET: Load balance message
received from TPS

oam> 07/11/03 13:54:05 LOG0004 CSV: ITS4011 47.11.215.230
connecting to node 3556, TN: 61-05, 0x6045 [Last Reset
Reason Not Supported, code:  127] (201)
07/11/03 13:54:10 LOG0006 VTM: ITS5008 Terminal connection
status:  47.11.215.230 ok (20)
07/11/03 13:54:10 LOG0006 SET: 47.11.215.230 TN 61-05
Registered with CS
07/11/03 13:54:10 LOG0004 CSV: ITS4011 47.11.215.136
connecting to node 3556, TN: 61-01, 0x6005 [Last Reset
Reason Not Supported, code:  127] (201)
07/11/03 13:54:15 LOG0006 VTM: ITS5008 Terminal connection
```

```
status:  47.11.215.136 ok (20)
07/11/03 13:54:15 LOG0006 SET: 47.11.215.136 TN 61-01
Registered with CS
```

## logConsoleOff, logConsoleOn

Syntax:

**`logConsoleOff`**
**`logConsoleOn`**

These commands control message printing to the VGMC Maintenance serial port. The commands do not apply to the Signaling Server. The current status of the Maintenance port logging is shown by the logShow command output for Console logging. The default is On.

The following example shows these commands in use.

```
VGMC> logConsoleOff
Console logging disabled
value = 25 = 0x19

VGMC> logConsoleOn
Console logging enabled
value = 24 = 0x18
```

## logFileOff, logFileOn

Syntax:

**`logFileOff`**
**`logFileOn`**

These commands control the log message output to the SYSLOG.n file. The commands do not apply to the Signaling Server. The current status of the file logging is shown by the logShow command output for File logging. The default is On.

## logPrintOff, logPrintOn

Syntax:

**`logPrintOff, logPrintOn`**

These commands control message printing to the active logon session. The commands do not apply to the Signaling Server. The active session is wherever the user has logged on to, that is, the serial port (console) or a Telnet session. The current status of the active session logging is shown by the logShow command output for TTY logging. The default is On.

The following example shows how only the active session is disabled when the user logs in through Telnet.

```
VGMC> logPrintOff
Log message printing disabled
value = 30 = 0x1e
VGMC>

VGMC> logPrintOn
Log message printing enabled
value = 29 = 0x1d
```

The following example demonstrates how both the console and active session logging are disabled when the user logs in through the VGMC serial port (console).

```
VGMC> logPrintOff
Console logging disabled
Log message printing disabled
value = 30 = 0x1e

VGMC> logPrintOn
Console logging enabled
Log message printing enabled
value = 29 = 0x1d
```

### logShow

Syntax:
**`logShow`**

Print the current status of the message printing for all tasks on the VGMC. Although you can run the command on the Signaling Server, it does not provide the same kind of information. Therefore, Nortel recommends that you use the syslogShow command instead.

When the Level is none, the task is not registered with the syslog function. The parameter Syslog file is the SYSLOG file used for output. Use this command with the syslogLevelSet command.

The following example shows command output on the ITG-P card.

```
-> logShow
TTY logging:  on
File logging:  off
Console logging:  on
Syslog file:  /C:/log/SYSLOG.0
Space remaining:  0
Excption file:  /C:/log/EXCPLOG.1
Space remaining:  0
```

```
Task Level
-------------------- -------
tExcTask none
tLogTask Info
tAioWait none
tAioIoTask1 none
tAioIoTask0 none
tPcmciad none
tTffsPTask none
tNetTask Info
tTelnetd none
tPortmapd none
tRdbTask none
tFtpdTask none
tTftpdTask none
tSnmpd none
tSyslogd Info
tMonTask none
tPxTimer none
tbootpd Info
tSNTPC Info
baseMMintTask none
tXA Info
tA07 Info
tRDP Info
tMAM Info
tMVX_XSPY Info
tMVX_DIM Info
tOMM Info
tRPCMGMT none
tELC Info
tVTM Info
tVTI Info
tSET Info
tCSV Info
tTPS Info
tfwBk Info
tUMS Info
tUMC Info
tVGW Info
tRTP Info
tRTCP Info
midnightTask none
tTelnetOutTask none
tTelnetInTask none
```

```
tyLstnr none
tShell none
value = 5 = 0x5
```

The following example shows command output on the SMC.

```
-> logShow
TTY logging:  on
File logging:  off
Console logging:  on
Syslog file:  /C:/log/syslog.3
Space remaining:  4651
Excption file:  /C:/log/excplog.0
Space remaining:  4096

Task Level
-------------------- -------
tExcTask none
tLogTask none
tAioWait none
tAioIoTask1 none
tAioIoTask0 none
tPcmciad none
tDcacheUpd none
tNetTask none
tTelnetd none
tPortmapd none
tFtpdTask none
tTftpdTask none
tSnmpd none
tSyslogd none
tMonTask none
tPxTimer none
tbootpd none
tSNTPC none
baseMMintTask none
tXA Info
tA07 Info
tRDP Info
tMAM Info
tMVX_XSPY none
tMVX_DIM Info
tOMM Info
tPBX Info
tELC Info
tVGW Info
tRTP Info
```

```
tRTCP Info
tXMSG Info
midnightTask none
tShell none
tTelnetOutTask none
tTelnetInTask none
tyLstnr none
value = 5 = 0x5
```

### lossPlanClr

Syntax:

**`lossPlanClr`**

Clears the gain adjustment made by the \UKLossPlanSet and lossPlanSet commands and returns the IP Phones to the TIA-810A levels. Any adjustments made to the handset, headset, or handsfree are cleared.

Enter this command on the node Leader card while it is the node master to ensure the data correctly propagates to all cards in the node. When you install a new Leader card on a node with modified levels, always enter the loss plan command on the CLI even if the command was previously entered on the CLI for another card.

```
VGMC> lossPlanClr
value = 0 = 0x0
VGMC>
IP client loss plan set to default values
```

### lossPlanPrt

Syntax:

**`lossPlanPrt`**

Displays the current loss plan settings for the IP Phones. Three columns are displayed. The Default column shows the design default in the IP Phones. The Offset column shows the current adjustment entered through the UKLossPlanSet or lossPlanSet commands. The Result column shows the resulting loss levels in the IP Phones.

```
VGMC> lossPlanPrt
Parameter Default Offset Result
--------------- ------- ------ ------
HandsetRLR 2 0 2
HandsetSLR 11 -5 6
HeadsetRLR 0 0 0
HeadsetSLR 11 -5 6
HandsfreeRLR 13 0 13
HandsfreeSLR 16 0 16
```

## lossPlanSet

Syntax:

**`lossPlanSet "transducer", rlrOffset, slrOffset`**

Change the gain settings on the IP Phone from the volume levels specified by the TIA-810A and TIA-912. The parameter transducer specifies the transducer name that requires a gain adjustment: handset, headset, or handsfree. The parameter rlrOffset specifies the amount of receive-level adjustment (the level for the IP Phone user), while the parameter slrOffset specifies the amount of transmit level adjustment (the level for the far-end user). Values for both are in the range –8 to 8 (in dB). Negative values (–8 to –1) increase the volume (gain).

Enter the command once for each transducer level requiring adjustment. Adjustments are not cumulative; the last one entered for a particular transducer replaces prior offset for the transducer.

Enter this command on the node Leader card while it is the node master to ensure that the data ipropagates correctly to all cards in the node. When you install a new leader card on a node with modified levels, always enter the loss plan command on the CLI even if the command was previously entered on the CLI on another card.

After you enter this command, the gain adjustment is downloaded to all registered telephones. When phones register, they are downloaded the new gain values. The gain adjustment is also saved to a disk file (/c:/config/loss.ini) so the adjusted gains are retained when the card reboots.

When a node has a modified loss plan (that is, when you use this command or UKLossPlanSet), a new card added to the node is updated with the modified loss plan 30 seconds after it boots. Prior to that being received, calls made by IP Phones registered to the new card have the default loss plan levels.

> ***Note:*** If the gain of the IP Phones is raised from the default levels, those phones can experience increased occurrence of echo and other audio issues related to the increased volume.

```
VGMC> lossPlanSet "headset",0,-3
value = 6 = 0x6
VGMC>
Headset loss changed to RLR = 0 SLR = 8
```

## mac21440BroadcastShow

Syntax:

**`mac21440BroadcastShow`**

Displays the sum of broadcast IP packets received each second on an SMC ELAN and TLAN interfaces for the past 60 seconds. This command is available only on the SMC. This output is the combined count for the ELAN and TLAN. The total number of broadcast messages received on the interface since the card was booted also appears.

```
-> mac21440BroadcastShow
Broadcasts per second, last 60 second
-0:  6
-1:  36
-2:  6
-3:  43
-4:  43
-5:  8
-6:  20
-7:  10
-8:  10
-9:  14
-10:  14
-11:  16
-12:  30
-13:  10
-14:  6
-15:  18
-16:  6
-17:  10
-18:  8
-19:  8
-20:  8
-21:  4
-22:  12
-23:  12
-24:  18
-25:  8
-26:  10
-27:  20
-28:  16
-29:  18
-30:  20
-31:  28
-32:  41
-33:  41
-34:  26
-35:  41
-36:  8
-37:  12
-38:  12
```

```
-39:  24
-40:  42
-41:  34
-42:  41
-43:  28
-44:  10
-45:  8
-46:  6
-47:  10
-47:  1342177296
-48:  40
-49:  28
-50:  20
-51:  18
-52:  28
-53:  12
-54:  8
-55:  4
-56:  8
-57:  32
-58:  28
-59:  18
Total number of broadcasts:  2260970
value = 36 = 0x24 = '$'
```

### mac21440BroadcastThreshold = threshold

This variable applies to the SMC only. Assigning this variable controls the threshold (in packets per second) at which a message prints when the mac21440ShowLostBroadcast variable is assigned. The default threshold is 40. Unlike the separate ITG-P Isa/Pci threshold commands, the sum of the broadcast traffic on the SMC ELAN and TLAN interfaces is checked against this threshold.

```
-> mac21440BroadcastThreshold = 100
mac21440BroadcastThreshold = 0x345604:  value = 100 = 0x64
= 'd'
```

### mac21440ShowLostBroadcast = 1/0

This variable applies to the SMC only. Assigning a value of 1 prints a message whenever the broadcast threshold (assigned using the mac21440BroadcastThreshold variable) is exceeded by the sum of the broadcasts on the SMC ELAN and TLAN interfaces. The default is off (0); the state is not saved and returns to off if the card is reset or reboots. The example shown was generated when the threshold was set to the default of 40.

```
-> mac21440ShowLostBroadcast = 1
mac21440ShowLostBroadcast = 0x34555c:  value = 1 = 0x1
->
NOV 26 14:10:40 tLogTask:  Info Time 104457 mac21440:  41
broadcasts in last second
NOV 26 14:10:44 tLogTask:  Info Time 104461 mac21440:  43
broadcasts in last second
NOV 26 14:11:00 tLogTask:  Info Time 104477 mac21440:  44
broadcasts in last second
NOV 26 14:11:05 tLogTask:  Info Time 104482 mac21440:  41
broadcasts in last second
NOV 26 14:11:09 tLogTask:  Info Time 104486 mac21440:  41
broadcasts in last second
NOV 26 14:11:10 tLogTask:  Info Time 104487 mac21440:  41
broadcasts in last second
NOV 26 14:11:11 tLogTask:  Info Time 104488 mac21440:  42
broadcasts in last second
NOV 26 14:11:15 tLogTask:  Info Time 104492 mac21440:  44
broadcasts in last second
```

## mbufShow

Syntax:

**`mbufShow`**

Display statistics and the distribution of the low-level buffers used by the IP stack.

```
-> mbufShow
type number
--------- ------
FREE : 6400
DATA : 0
HEADER : 0
SOCKET : 0
PCB : 0
RTABLE : 0
HTABLE : 0
ATABLE : 0
SONAME : 0
ZOMBIE : 0
SOOPTS : 0
FTABLE : 0
RIGHTS : 0
IFADDR : 0
CONTROL : 0
OOBDATA : 0
IPMOPTS : 0
```

```
IPMADDR : 0
IFMADDR : 0
MRTABLE : 0
TOTAL : 6400
number of mbufs:  6400
number of times failed to find space:  0
number of times waited for space:  0
number of times drained protocols for space:  0
__________________
CLUSTER POOL TABLE
_______________________________________________________
________________________
size clusters free usage
-------------------------------------------------------
------------------------
64 1024 1024 1756686
128 1024 1024 2890311
256 512 512 2331
512 512 512 3
1024 512 512 0
2048 512 512 0
-------------------------------------------------------
------------------------
value = 80 = 0x50 = 'P'
```

The following example shows the output on an idle Signaling Server.

```
->mbufShow
type number
--------- ------
FREE : 36858
DATA : 4
HEADER : 2
SOCKET : 0
PCB : 0
RTABLE : 0
HTABLE : 0
ATABLE : 0
SONAME : 0
ZOMBIE : 0
SOOPTS : 0
FTABLE : 0
RIGHTS : 0
IFADDR : 0
CONTROL : 0
OOBDATA : 0
IPMOPTS : 0
```

```
IPMADDR : 0
IFMADDR : 0
MRTABLE : 0
TOTAL : 36864
number of mbufs:  36864
number of times failed to find space:  0
number of times waited for space:  0
number of times drained protocols for space:  0
__________________
CLUSTER POOL TABLE
_______________________________________________________
________________________
size clusters free usage
-------------------------------------------------------
------------------------
64 4096 4092 149613
128 4096 4096 178567
256 4096 4096 2350
512 4096 4096 18
1024 1024 1024 452
2048 1024 1024 0
-------------------------------------------------------
------------------------
value = 80 = 0x50 = 'P'
```

## memShow

Syntax:

**memShow**

Display the following memory statistics on the Signaling Server:

- free memory and the blocks that it is composed of
- memory in use
- cumulative amount of memory allocated

The command itgMemShow prints the same data as memShow.

```
-> memShow
status bytes blocks avg block max block
------ --------- -------- ---------- ----------
current
free 72563664 135 537508 72261852
alloc 60945568 18235 3342 -
cumulative
alloc 1036327240 1444501 717 -
value = 0 = 0x0
```

## memShowPatch

Syntax:

**`memShowPatch`**

This command displays the memory statistics for the patch memory partition on the SMC card, including:

- free memory and the blocks that it is composed of
- memory in use
- cumulative amount of memory allocated

```
-> memShowPatch
status bytes blocks avg block max block
------ --------- -------- ---------- ----------
current
free 3122496 1 3122496 3122496
alloc 23216 3 7738 -
cumulative
alloc 46024 4 11506 -
value = 0 = 0x0
```

## mkdir

Syntax:

**`mkdir "dirName"`**

Creates a new subdirectory of the current directory.

```
-> ll
size date time name
-------- ------ ------ -------- 512 JUN-06-2000 08:28:44
LOG <DIR>
512 JUN-06-2000 08:28:46 OM <DIR>
512 JUN-06-2000 08:28:50 CONFIG <DIR>
512 JUN-06-2000 08:33:34 FW <DIR>
512 JAN-01-1996 12:00:22 DATA <DIR>
512 JAN-01-1996 12:00:22 ETC <DIR>
512 JAN-01-1996 12:00:22 LOCALE <DIR>
value = 0 = 0x0
-> mkdir "temp"
value = 0 = 0x0
-> ll
size date time name
-------- ------ ------ -------- 512 JUN-06-2000 08:28:44
LOG <DIR>
512 JUN-06-2000 08:28:46 OM <DIR>
512 JUN-06-2000 08:28:50 CONFIG <DIR>
512 JUN-06-2000 08:33:34 FW <DIR>
```

```
512 JAN-01-1996 12:00:22 DATA <DIR>
512 JAN-01-1996 12:00:22 ETC <DIR>
512 JAN-01-1996 12:00:22 LOCALE <DIR>
512 FEB-22-2001 15:10:00 TEMP <DIR>
value = 0 = 0x0
```

## mRouteAdd

Syntax:

**`mRouteAdd "destIPaddr", "gwIPaddr", 0xdestNetMask, ToS, 0`**

This command is similar to routeAdd, but allows multiple routes to the same destination, differentiated by the ToS or gateway fields. The parameter destIPaddr is the destination IP address; gwIPaddr is the gateway IP address; 0xdestNetMask is the destination net mask in hexadecimal format; ToS is the Type of Service for this route. This change is temporary; rebooting the card rebuilds the routing table from the data in the CONFIG.INI file.

```
-> mRouteAdd "192.168.1.128","192.168.1.140",0xffffff80,
0,0
value = 0 = 0x0
```

## mRouteDelete

Syntax:

**`mRouteDelete "destIPaddr", 0xdestNetMask, ToS`**

This command is similar to routeDelete, but specifies the route using the destination address, netmask and ToS. The parameter destIPaddr is the destination IP address; 0xdestNetMask is the destination net mask in hexadecimal format; ToS is the Type of Service for this route. This change is temporary; rebooting the card rebuilds the routing table from the data in the CONFIG.INI file.

```
-> mRouteDelete "192.168.1.128",0xffffff80,0
value
```

## mRouteShow

Syntax:

**`mRouteShow`**

Similar to routeShow, but also displays the ToS bit and mask settings.

```
-> mRouteShow
Destination Mask TOS Gateway Flags RefCnt Use Interface
Proto
0.0.0.0 0 0 192.168.1.200 3 0 0 lnPci1 0
127.0.0.1 0 0 127.0.0.1 5 0 0 lo0 0
```

```
192.168.1.0 ffffff80 0 192.168.1.14 101 0 0 lnIsa0 0
192.168.1.128 ffffff80 0 192.168.1.140 101 0 0 lnPci1 0
192.168.1.128 ffffff80 18 192.168.1.140 101 0 0 lnPci1 1
value = 0 = 0x0
```

## netHelp

Syntax:

**`netHelp`**

Display a help screen synopsis of often used network routines.

The following is an example of the output, which is the same on the ITG-P, SMC or Signaling Servers.

```
-> netHelp (itg P2 , SA and SS the same)
hostAdd "hostname","inetaddr" - add a host to remote host
table;
"inetaddr" must be in standard Internet address format e.g.
"90.0.0.4"
hostShow - print current remote host table
netDevCreate "devname","hostname",protocol - create an
I/O device to access files on the specified host (protocol
0=rsh, 1=ftp) routeAdd "destaddr","gateaddr" - add route to
route table
routeDelete "destaddr","gateaddr" - delete route from
route table
routeShow - print current route table
iam "usr"[,"passwd"] - specify the user name by which you
will be known to remote hosts(and optional password)
whoami - print the current remote ID rlogin
"host" - log in to a remote host; "host" can be inet address
or host name in remote host table

Type <CR>to continue, Q<CR>to stop:

ifShow ["ifname"] - show info about network interfaces
inetstatShow - show all Internet protocol sockets
tcpstatShow - show statistics for TCP
udpstatShow - show statistics for UDP
ipstatShow - show statistics for IP
icmpstatShow - show statistics for ICMP

arptabShow - show a list of known ARP entries
mbufShow - show mbuf statistics

EXAMPLE: -> hostAdd "wrs", "90.0.0.2"
-> netDevCreate "wrs:", "wrs", 0
-> iam "fred"
-> copy <wrs:/etc/passwd /* copy file from host "wrs"
*/
-> rlogin "wrs" /* rlogin to host "wrs" */

value = 1 = 0x1
```

## nodePwdEnable, nodePwdDisable

Syntax:

**nodePwdEnable**
**nodePwdDisable**

Enable or disable node password checking.

The following example is the output for the VGMC but the same data prints on the Signaling Server.

```
VGMC> nodePwdDisable
value = 0 = 0x0

VGMC> nodePwdShow
value = 0 = 0x0

NodeID PwdEna Pwd TmpPwd Uses TimeOut
====== ====== ============== ============== ==========
=================
1 No 0 0d
0h 0m 0s

VGMC> nodePwdEnable
value = 0 = 0x0

VGMC> nodePwdShow
value = 0 = 0x0

NodeID PwdEna Pwd TmpPwd Uses TimeOut
====== ====== ============== ============== ==========
=================
1 Yes 0 0d
0h 0m 0s
```

## nodePwdSet

Syntax:

**nodePwdSet**

Assign the node password. If you enter a non-zero length password, all IP Phones that attempt to register prompts the user for a node password before the TN can be modified.

The following example is the output for the VGMC, but the same data prints on the Signaling Server.

```
nodePwdSet - with password:  VGMC> nodePwdSet "1234567890"
value = 0 = 0x0 VGMC> nodePwdShow value = 0 = 0x0 NodeID
PwdEna Pwd TmpPwd Uses TimeOut ====== ====== ==============
============== ========== ================== 1 Yes
1234567890 0 0d 0h 0m 0s nodePwdSet -without password:
VGMC> nodePwdSet "" value = 0 = 0x0 VGMC> nodePwdShow
value = 0 = 0x0 VGMC> NodeID PwdEna Pwd TmpPwd Uses TimeOut
====== ====== ============== ============== ==========
================== 1 Yes 0 0d 0h 0m 0s
```

## nodePwdShow

Syntax:
**`nodePwdShow`**

Print the node password settings.

The following example is the output for the VGMC, but the same data prints on the Signaling Server.

```
VGMC> nodePwdShow value = 0 = 0x0 VGMC> NodeID PwdEna
Pwd TmpPwd Uses TimeOut ====== ====== ==============
============== ========== ================== 1 Yes
1234567890 0 0d 0h 0m 0s
```

## nodeTempPwdClear, nodeTempPwdSet

Syntax:
**`nodeTempPwdClear "Password", "Uses", "Timeout"`**
**`nodeTempPwdSet "Password", "Uses", "Timeout"`**

The nodeTempPwdSet command sets the node level TN entry temporary password. The nodeTempPwdClear command deletes the temporary password and resets its uses and time to zero.

The following example is the output for the VGMC, but the same data prints on the Signaling Server.

```
VGMC> nodeTempPwdSet "1234567", 20, 1 value = 0 = 0x0 VGMC>
nodePwdShow value = 0 = 0x0 VGMC> NodeID PwdEna Pwd TmpPwd
Uses TimeOut ====== ====== ============= =============
===== ================= 555 Yes 12345678 1234567 20 0d 1h
0m 0s VGMC>
```

One telephone registers using the temporary password. The Timeout and Uses values decrease.

```
JUL 10 04:04:01 tCSV: Info 192.168.20.100 Connecting
to node555, TN:61.23, 0x6147 JUL 10 04:04:08 tVTM: Info
ITS5008 Terminal connection status:  192.168.20.100 ok
(20) JUL 10 04:04:08 tSET: Info 192.168.20.100 TN 61-23
Registered with CS VGMC> nodePwdShow value = 0 = 0x0
VGMC> NodeID PwdEna Pwd TmpPwd Uses TimeOut ====== ======
============= ============= ===== =================
555 Yes 12345678 1234567 19 0d 0h 58m 20s VGMC> VGMC>
nodeTempPwdClear value = 0 = 0x0 VGMC> nodePwdShow value
= 0 = 0x0 VGMC> NodeID PwdEna Pwd TmpPwd Uses TimeOut
====== ====== ============= ============= =====
================= 555 Yes 12345678 0d 0h 0m 0s
```

## NVRClear

Syntax:

**`NVRClear`**

Erase the VGMC ELAN IP data and Leader flag and returns the shell logon and shell timeout to the default values. This command applies only to the VGMCs. Use caution with this command because the card appears as a Follower, but you do not require a BOOTP request. Use the clearLeader command to return a Leader card to Follower status or the command shellTi to reset the shell password. If you use the NVRClear command to clean up the NVRAM, follow it with the clearLeader or setLeader command.

```
VGMC> NVRIPShow IP address :  192.168.1.14 Gateway :
192.168.1.1 Subnet Mask:  255.255.255.128 Set as Leader.
Set to use IP address parameters in NVRAM. value = 43 = 0x2b =
'+' VGMC> NVRClear value = 0 = 0x0 VGMC> NVRIPShow IP address
:  255.255.255.255 Gateway :  0.0.0.0 Subnet Mask:  0.0.0.0
Set as Follower.  Warning :  Set as Follower, but will boot
using IP parameters in NVRAM instead of bootp.  value = 88 =
0x58 = 'X'
```

## NVRGWSet

Syntax:

**`NVRGWSet "gwIPaddr"`**

Assign the card ELAN gateway IP address in the NVRAM. This command applies only to the VGMCs.

### NVRIPSet

Syntax:

**`NVRIPSet "IPaddr"`**

Assign the card ELAN IP address in the NVRAM. This command applies only to the VGMCs.

### NVRIPShow

Syntax:

**`NVRIPShow`**

Print the information programmed in the card NVRAM. This command applies only to the VGMCs. This command is useful to determine whether the data configured using the setLeader command is correct on the card.

```
-> NVRIPShow IP address :  192.168.1.14 Gateway :
192.168.1.1 Subnet Mask:  255.255.255.128 Set as Leader.
Set to use IP address parameters in NVRAM. value = 43 = 0x2b =
'+'
```

### NVRSMSet

Syntax:

**`NVRSMSet "subnetMask"`**

Assign the card ELAN subnet mask in the NVRAM. This command applies only to the VGMCs.

### ommShow

Syntax:

**`ommShow`**

Display the current value of OM counters. Enter this command to view the OM statistics; it does not affect the counter values or the content of the OMREPORT.nnn file.

```
-> ommShow collection_time :  2/23/2001 5:03 i2004Reg_Att:
0 i2004Reg_Fail:  0 i2004Unreg_Att:  0 i2004Aud_Setup:  4
i2004Jitter_Avg:  3.7 i2004Jitter_Max:  13 i2004Pkt_Lost:
0.00 i2004Voice_Time:  1 mins 20 secs ChanAud_Setup:  4
ChanJitter_Avg:  6.0 ChanJitter_Max:  52 ChanPkt_Lost:
0.00 ChanVoice_Time:  1 mins 20 secs value = 271 = 0x10f
```

## osClockShow

Syntax:

**`osClockShow`**

Display the date and time as known by the VxWorks OS clock.

```
-> osClockShow
OS Time:  Date (12/02/2001) Time (16:42:31)
value = 44 = 0x2c = ','
```

## pbxLibResetLink

Syntax:

**`pbxLibResetLink`**

Reset the link between the VGMC/Signaling Server and the Call Server. The link is released and link reestablishment attempted.

The following example shows the SMC output (the trace for the ITG-P is identical except for the number of gateway channels that register).

```
-> pbxLibResetLink
value = 35970160 = 0x224dc70
-> 47.11.216.184 (47.11.216.184) deleted-> 47.11.216.184
(47.11.216.184) deleted

OCT 31 10:24:56 tpbxResetMain:  Alert ITS1009 Call server
communication link:  47.11.216.184 down (20)
OCT 31 10:24:56 tpbxResetMain:  Warning Elan link down,
call election after 2 minutes
OCT 31 10:24:56 tpbxResetMain:  Info pbxTcpLinkStop:  send
shutdown msg to tcp recv task
OCT 31 10:24:56 tpbxResetMain:  Info pbxTcpLinkClose:
Close the tcp socket
OCT 31 10:24:56 tPBX: Info TCP close msg received
OCT 31 10:24:56 tpbxResetMain:  Info TCP msg pipe closed
OCT 31 10:24:56 tpbxResetMain:  Info Attempting to
establish PBX link with link 0
OCT 31 10:24:56 TPS: Info ELAN connection down, refuse
further registration
OCT 31 10:24:56 VGW: Info GW OffLine msg received from
pbxLib, close all dsp channels and unregister gateways
after 600 seconds OCT 31 10:24:56 VGW: Info Flushing
registration queue because the ELAN link is down
OCT 31 10:24:56 SET: Info PBX Link down, reset all
registered set after 600 seconds
OCT 31 10:24:56 SET: Info Flushing 0 DSET regsitration
requests because ELAN link is down
```

```
OCT 31 10:24:56 SET: Info Flushing 0 keymap download
requests becauses of ELAN link down
->
->
OCT 31 10:25:11 tpbxResetMain:  Info Sending request msg to
get Call Server S/W information
OCT 31 10:25:31 tpbxResetMain:  Info Sending request msg to
get Call Server S/W information
OCT 31 10:25:31 RDP: Info Parsing the call server
supportability information
OCT 31 10:25:32 tpbxResetMain:  Info PBX UDP link
established for link 0
OCT 31 10:25:32 tpbxResetMain:  Info Successfully got call
server information
OCT 31 10:25:32 tpbxResetMain:  Info TCP msg pipe to
47.11.216.184:15000 established
OCT 31 10:25:32 tPBX: Info TCP msg read task started...
OCT 31 10:25:32 tpbxResetMain:  Info PBX TCP link
established OCT 31 10:25:32 tpbxResetMain:  Info ITS5009
Call server communication link:  47.11.216.184 up (20)
OCT 31 10:25:32 TPS: Info ELAN connection up, accept set
registration
OCT 31 10:25:32 VGW: Info GW OnLine msg received from pbxLib
OCT 31 10:25:32 VGW: Info vgwSyncReqReceive:  freq 0,
rlsCall 1, callServer 2 OCT 31 10:25:32 VGW: Info sending TN
call Proc status request for VGW
OCT 31 10:25:32 VGW: Info Registering 32 channels
OCT 31 10:25:32 MAM: Info itgMsgPBXTimeRequest send:
Day/Month/Year 31/10/2002, hrs/min/sec 10/25/32
OCT 31 10:25:32 CSV: Info call server using the short TN
format
OCT 31 10:25:32 CSV: Info call server using the short TN
format
OCT 31 10:25:32 SET: Info dsetSyncReqReceive:  freq 0,
rlsCall 1, callServer 2
OCT 31 10:25:32 SET: Info sending TN call Proc status
request for DSET
OCT 31 10:25:32 VGW: Info Channel 0, already registered with
CS (?)
OCT 31 10:25:32 VGW: Info Channel 1, already registered with
CS (?)
OCT 31 10:25:32 VGW: Info Channel 2, already registered with
CS (?)
OCT 31 10:25:32 VGW: Info Channel 3, already registered with
CS (?)
OCT 31 10:25:32 VGW: Info Channel 4, already registered with
CS (?)
```

```
OCT 31 10:25:32 SET: Info tone table replaced
OCT 31 10:25:32 VGW: Info Channel 5, already registered with
CS (?)
OCT 31 10:25:32 VGW: Info Channel 6, already registered with
CS (?)  OCT 31 10:25:32 SET: Info cadence table replaced
OCT 31 10:25:32 SET: Info server <vxTarget> node <7812>
online announce
OCT 31 10:25:32 VGW: Info Channel 7, already registered with
CS (?)
OCT 31 10:25:32 SET: Info ServerStatus OK
OCT 31 10:25:32 VGW: Info Channel 8, already registered with
CS (?)
OCT 31 10:25:32 VGW: Info Channel 9, already registered with
CS (?)
OCT 31 10:25:32 VGW: Info Channel 10, already registered
with CS (?)
OCT 31 10:25:32 VGW: Info Channel 11, already registered
with CS (?)
OCT 31 10:25:32 VGW: Info Channel 12, already registered
with CS (?)
OCT 31 10:25:32 VGW: Info Channel 13, already registered
with CS (?)
OCT 31 10:25:32 VGW: Info Channel 14, already registered
with CS (?)
OCT 31 10:25:32 VGW: Info Channel 15, already registered
with CS (?)
OCT 31 10:25:32 VGW: Info Channel 16, already registered
with CS (?)
OCT 31 10:25:32 VGW: Info Channel 17, already registered
with CS (?)
OCT 31 10:25:32 VGW: Info Channel 18, already registered
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 19, already registered
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 20, already registered
with CS (?)
OCT 31 10:25:33 tNetTask:  Info mac21440Send:  Pkt 64 byte
multiple.  tfifosRequired = 21
OCT 31 10:25:33 VGW: Info Channel 21, already registered
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 22, already registered
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 23, already registered
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 24, already registered
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 25, already registered
```

```
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 26, already registered
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 27, already registered
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 28, already registered
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 29, already registered
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 30, already registered
with CS (?)
OCT 31 10:25:33 VGW: Info Channel 31, already registered
with CS (?)
OCT 31 10:25:35 ELC: Notice Election won, master =
192.168.20.5
OCT 31 10:25:35 TPS: Info Security Check is Enabled
```

The following example shows command output on the Signaling Server.

```
-> pbxLibResetLink
value = 176095708 = 0xa7f01dc
-> 47.11.216.184 (47.11.216.184) deleted
31/10/02 10:27:15 LOG0001 tpbxResetMain:  ITS1009 Call
server communication link:  47.11.216.184 down (20)
31/10/02 10:27:15 LOG0003 tpbxResetMain:  itgMsgSend to
task 0xbc00
31/10/02 10:27:15 LOG0004 tpbxResetMain:  Elan link down,
call election after 2 minutes
31/10/02 10:27:15 LOG0006 tpbxResetMain:  pbxTcpLinkStop:
send shutdown msg to tcp recv task
31/10/02 10:27:15 LOG0006 tpbxResetMain:  pbxTcpLinkClose
:  Close the tcp socket
31/10/02 10:27:15 LOG0006 tPBX: TCP close msg received
31/10/02 10:27:15 LOG0006 tpbxResetMain:  TCP msg pipe
closed
31/10/02 10:27:15 LOG0006 tpbxResetMain:  Attempting to
establish PBX link with link 0
31/10/02 10:27:15 LOG0006 TPS: ELAN connection down, refuse
further registration
31/10/02 10:27:15 LOG0006 SET: PBX Link down, reset all
registered set after 600 seconds
31/10/02 10:27:15 LOG0006 SET: Flushing 0 DSET regsitration
requests because ELAN link is down
31/10/02 10:27:15 LOG0006 SET: Flushing 0 keymap download
requests becauses of ELAN link down
31/10/02 10:27:30 LOG0006 tpbxResetMain:  Sending request
msg to get Call Server S/W information
```

```
31/10/02 10:27:50 LOG0006 tpbxResetMain:  Sending request
msg to get Call Server S/W information
31/10/02 10:27:50 LOG0006 RDP: Parsing the call server
supportability information
31/10/02 10:27:50 LOG0006 RDP: Companding Law set to MuLaw
31/10/02 10:27:50 LOG0006 RDP: notifyConfigReadDone for 0
channels
31/10/02 10:27:51 LOG0006 tpbxResetMain:  PBX UDP link
established for link 0
31/10/02 10:27:51 LOG0006 tpbxResetMain:  Successfully got
call server information
31/10/02 10:27:51 LOG0006 tpbxResetMain:  TCP msg pipe to
47.11.216.184:15000 established
31/10/02 10:27:51 LOG0006 tPBX: TCP msg read task
started...
31/10/02 10:27:51 LOG0006 tpbxResetMain:  PBX TCP link
established
31/10/02 10:27:51 LOG0003 tpbxResetMain:  itgMsgSend to
task 0xbc00
31/10/02 10:27:51 LOG0006 tpbxResetMain:  ITS5009 Call
server communication link:  47.11.216.184 up (20)
31/10/02 10:27:51 LOG0003 tpbxResetMain:  itgMsgSend to
task 0xbc00
31/10/02 10:27:51 LOG0006 TPS: ELAN connection up, accept
set registration
31/10/02 10:27:51 LOG0006 MAM: itgMsgPBXTimeRequest send:
Day/Month/Year
31/10/2002, hrs/min/sec
10/27/51 31/10/02 10:27:51 LOG0006 CSV: call server using
the short TN format
31/10/02 10:27:51 LOG0006 SET: dsetSyncReqReceive:  freq
0, rlsCall 1, callServer 2
31/10/02 10:27:51 LOG0006 SET: sending TN call Proc status
request for DSET 31/10/02
10:27:51 LOG0006 MAM: system time synchronized with call
server time:  THU OCT 31 10:27:51 2002

31/10/02 10:27:51 LOG0006 SET: tone table replaced
31/10/02 10:27:51 LOG0006 SET: cadence table replaced
31/10/02 10:27:51 LOG0006 SET: server <jimfan> node
<7812>online announce
31/10/02 10:27:51 LOG0006 SET: ServerStatus OK
```

### pbxLinkShow

Syntax:

```
pbxLinkShow
```

Displays the status of the link with the Call Server (Link state), the Call Server ELAN IP address and the protocol used for the communication.

The VGMC and Signaling Servers communicate with the Call Server using TCP. Even when you use TCP, the heartbeat between the VGMC/Signaling Server and the Call Server occurs over RUDP.

```
-> pbxLinkShow
Active CS type = Succession CSE 1K
Active CS S/W Release = 300
Supported Features:  CorpDir UserKeyLabel VirtualOffice
UseCSPwd I2001 I2004 Ph2 I2002 Ph2
CS Main:  ip = 47.11.216.85, ConnectID = 0xb9ddc34,
BroadcastID = 0xb9ddd30, Link is up
CS Signaling Port = 15000
CS Broadcast Port = 15001
Broadcast PortID = 0xb9ddeb8
RUDP portID = 0xb9dde2c
Tcp Link state = up Tcp
Signaling Port:  15000
Tcp socket fd:  40
Tcp msgs sent:  175248
Tcp msgs recd:  2319379
value = 23 = 0x17
```

## ping

Syntax:
**`ping "IPaddr", [numPings]`**

This command sends an ICMP ECHO_REQUEST packets to a network host, specified by the parameter IPaddr in dotted notation. The host that matches the destination address in the packets responds to the request. If a response is not returned in less than 5 seconds, the sender times out. You can use this command to determine whether other hosts or VGMCs are properly communicating with the sender card. The command also works on the Signaling Server.

The parameter numPings specifies the number of packets to receive; if omitted, pings continue until you type Ctrl+C. Otherwise, pings are sent until the specified number of responses are received. The percentage of packet loss is then calculated from the ratio of received to sent messages.

```
VGMC> ping "10.0.0.2",5
PING 10.0.0.2:  56 data bytes
64 bytes from 10.0.0.2:  icmp_seq=0.  time=0.  ms
64 bytes from 10.0.0.2:  icmp_seq=1.  time=0.  ms
64 bytes from 10.0.0.2:  icmp_seq=2.  time=0.  ms
```

```
64 bytes from 10.0.0.2:  icmp_seq=3.  time=0.  ms
64 bytes from 10.0.0.2:  icmp_seq=4.  time=0.  ms
----10.0.0.2 PING Statistics----
5 packets transmitted, 5 packets received, 0% packet loss
round-trip (ms) min/avg/max = 0/0/0
value = 0 = 0x0
```

## pdPipeShow

Syntax:
**pdPipeShow**

This command displays the detail configuration and run-time data for the smart pipe.

```
-> pdPipeShow Smart Pipe [PD Smart Pipe] --------------
Run Time Data ------------- Pending Pipe Reqs .....  0
HTTP Req pending ......  no Pipe Timer active .....  no
Peak ..................  2 Maximum RTT ...........
384 (ms) Pipe Enabled ..........  yes Pipe Error
Count ......  0 Comm Error Count ......  0 <-- this
parameter indicates how many times has the SPIPE failed
communicating with the application server -------
Configuration ------------------ Application ...........
http://47.11.217.18/cgi/pdpipe.cgi Default timeout
......  250 (ms) Maximum timeout ......  20000 (ms) Maximum
Elements ......  44000 (reqs) Maximum Resp Size......
20000 (bytes) Maximum Req one shot ..  100 (reqs)
```

## pdWebCount

Syntax:
**pdWebCount**

Print the hit count for all major CGIs.

```
-> pdWebCount pdperdir.cgi ......  6780 pdcaller.cgi
......  7890 pdredial.cgi ......  2322 pdchgmod.cgi ......
1245
```

## pwd

Syntax:
**pwd**

Show the current directory path. On the VGMC, if the path shown is only host, you must change to the C: drive to make the other directories visible.

```
-> pwd
host:  value = 6 = 0x6
-> cd "/C:"
value = 0 = 0x0
-> pwd
/C:
value = 4 = 0x4
```

The result is different on the Signaling Server because of the difference in directory structures.

```
-> pwd
/u/config
value = 10 = 0xa
```

## rdxxxx commands (Signaling Server only)

The Signaling Server supports the same report log mechanism employeed on the CS 1000. Instead of using SYSLOG.n files as on the VGMCs, the information, warning, and error messages are stored in RPT files. You can access these files using the same rdxxxx commands used from the PDT shell on the CS 1000.

To print a list of commands, enter the following commands:

```
pdt> help rdtools
rdtools
Help for rd tools:

rdopen Open a report log file
rdgo Go to a specific record
rd Display records
rds Display records with symbolic dump
rdshow Show general log file info
rdall Display all records
rdtail Display newest records
rdhead Display oldest records
pdt>
```

The following example shows command output.

```
pdt> rdshow
File Name :  "/u/rpt/LOG00012.RPT" Capacity in bytes :
1000000 Capacity in records :  1315 Number of records =
1315 Oldest record = 0, logged at 26/11/02 04:49:21 Newest
record = 1314, logged at 26/11/02 17:08:22 Current Record =
0 Display Increment = 10 records
```

```
pdt> rdtail
[1314] 26/11/02 17:08:22 LOG0006 tRootTask:  Task
umsClientStubInit initialization succeeded
[1313] 26/11/02 17:08:21 LOG0006 VTM: VTM received
notification that UMC is initialized.
[1312] 26/11/02 17:08:21 LOG0006 CSV: Node 3918 registering
for terminal connections on 47.11.215.43:4100
[1311] 26/11/02 17:08:21 LOG0006 CSV: CSV enable
[1310] 26/11/02 17:08:21 LOG0006 CSV: UMC is initialized.
Enable CSV from pending.
[1309] 26/11/02 17:08:21 LOG0006 VTM: Client logged on
0xB13DD8C
[1308] 26/11/02 17:08:19 LOG0006 TPS: Firmware download for
Terminal i2002 completed.  Sending broadcast message from
IP:47.11.215.44
[1307] 26/11/02 17:08:19 LOG0006 tRootTask:  Task umsInit
initialization succeeded
[1306] 26/11/02 17:08:19 LOG0006 tRootTask:  Resync
IniFile from version 3003B20 to 0603B39 for terminal i2002.
[1305] 26/11/02 17:08:19 LOG0006 tRootTask:  Successfully
downloaded i2002 firmware from /u/fw/i2002.fw
pdt>
pdt> rdhead
[0000] 26/11/02 04:49:21 LOG0005 ELC: Election won, master
= 47.11.215.44
[0001] 26/11/02 04:49:21 LOG0006 TPS: Security Check is
Enabled
[0002] 26/11/02 04:49:21 LOG0006 TPS: Overriding password
to <123456>
[0003] 26/11/02 04:59:46 LOG0006 GKDBM: gkDbmMaintDynamicS
ync:  PRIMARY_GK in GK_ACTIVE
[0004] 26/11/02 04:59:46 LOG0006 GKDBM: gkDbmMaintDynamicS
ync:  Saved 0 Deleted 0
[0005] 26/11/02 04:59:49 LOG0006 GKOMM: Updating
/u/gk/omm/cur_omm.txt:  hour 19
[0006] 26/11/02 05:27:56 LOG0005 ELC: Election called by
47.11.215.30, reason <Timeout>
[0007] 26/11/02 05:27:58 LOG0006 ELC: VTRK: This signal
server is master
[0008] 26/11/02 05:27:58 LOG0006 ELC: gkNpmCardEventHandle
r:  unhandled event 0x4
[0009] 26/11/02 05:27:58 LOG0004 VTRK: vtrkStateHandler:
event 0 ignored ; state = VtrkActive
pdt>
```

## reboot

Syntax:

**`reboot type`**

Reboot the Signaling Server. This command does not apply to the VGMCs. When you leave the parameter type blank, the Signaling Server performs a warm start. When you assign it a value of –1, the Signaling Server performs a cold start.

## rename

Syntax:

**`rename "sourceFile", "destFile"`**

On the VGMC or Signaling Server, this command can be used to rename the file specified by sourceFilename to the name destFilename.

## resetOM

Syntax:

**`resetOM`**

Write the current OM counter values to the OMREPORT.nnn file and reset the counters to zero.

The following example is shown for a VGMC but also applies to the Signaling Server vxshell or oam prompts.

```
VGMC> resetOM
value = 0 = 0x0
VGMC>
FEB 23 05:06:58 tMAM: Info mamProcResetOm()
VGMC>
```

## rm

Syntax:

**`rm "filename"`**

Delete a file from the A: or C: drives. The A: drive is the faceplate PC card port on the VGMC, or the floppy drive on the Signaling Server.

```
-> ll
size date time name
-------- ------ ------ --------
512 JUN-06-2000 08:28:50 .  <DIR>
512 JUN-06-2000 08:28:50 ..  <DIR>
840 JAN-26-2001 13:00:38 CONFIG.INI
840 JAN-26-2001 13:17:40 CONFIG.BAK
238 JAN-23-2001 12:55:12 BOOTP.TAB
```

```
238 FEB-12-2001 17:26:12 BOOTP.BAK
205 FEB-12-2001 17:48:02 UMS.INI
21 NOV-14-2000 09:35:16 ITG.INI
value = 0 = 0x0
-> rm "ITG.INI"
value = 0 = 0x0
-> ll
size date time name
-------- ------ ------ --------
512 JUN-06-2000 08:28:50 .  <DIR>
512 JUN-06-2000 08:28:50 ..  <DIR>
840 JAN-26-2001 13:00:38 CONFIG.INI
840 JAN-26-2001 13:17:40 CONFIG.BAK
238 JAN-23-2001 12:55:12 BOOTP.TAB
238 FEB-12-2001 17:26:12 BOOTP.BAK
205 FEB-12-2001 17:48:02 UMS.INI
value = 0 = 0x0
```

### rmdir

Syntax:
**`rmdir "dirName"`**

Delete the specified directory.

```
-> ll
size date time name
-------- ------ ------ --------
512 JUN-06-2000 08:28:44 LOG <DIR>
512 JUN-06-2000 08:28:46 OM <DIR>
512 JUN-06-2000 08:28:50 CONFIG <DIR>
512 JUN-06-2000 08:33:34 FW <DIR>
512 JAN-01-1996 12:00:22 DATA <DIR>
512 JAN-01-1996 12:00:22 ETC <DIR>
512 JAN-01-1996 12:00:22 LOCALE <DIR>
512 FEB-22-2001 15:10:00 TEMP <DIR>
value = 0 = 0x0
-> rmdir "temp"
value = 0 = 0x0
-> ll
size date time name
-------- ------ ------ --------
512 JUN-06-2000 08:28:44 LOG <DIR>
512 JUN-06-2000 08:28:46 OM <DIR>
512 JUN-06-2000 08:28:50 CONFIG <DIR>
512 JUN-06-2000 08:33:34 FW <DIR>
512 JAN-01-1996 12:00:22 DATA <DIR>
```

```
512 JAN-01-1996 12:00:22 ETC <DIR>
512 JAN-01-1996 12:00:22 LOCALE <DIR>
value = 0 = 0x0
```

### routeAdd

Syntax:

**`routeAdd "destIPaddr", "gwIPaddr"`**

Add a route to the network routing tables. The parameter destIPaddr is the destination IP address; gwIPaddr is the gateway IP address. This change is temporary; rebooting the card rebuilds the routing table from the data in the CONFIG.INI file.

```
-> routeAdd "192.168.1.128","192.168.1.140"
value = 0 = 0x0
```

### routeDelete

Syntax:

**`routeDelete "destIPaddr", "gwIPaddr"`**

Deletes a route from the network routing tables. The parameter destIPaddr is the destination IP address in dotted notation; gwIPaddr is the gateway IP address in dotted notation. This change is temporary; rebooting the card rebuilds the routing table from the data in the CONFIG.INI file.

```
-> routeDelete "192.168.1.128","192.168.1.140"
value = 0 = 0x0
```

### routeShow

Syntax:

**`routeShow`**

Prints the card IP network and host routing tables. Verify the configured routes and the dynamic routes are in the table. The command also applies to the Signaling Server from the Signaling Server vxshell or oam prompts.

```
VGMC> routeShow
ROUTE NET TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
---------------------
0.0.0.0 10.0.0.1 3 1 302727 lnPci1
10.0.0.0 10.0.0.9 101 0 0 lnPci1
192.169.0.0 192.169.0.11 101 0 0 lnIsa0
-------------------------------------------------------
---------------------
```

```
ROUTE HOST TABLE
destination gateway flags Refcnt Use Interface
-------------------------------------------------------
---------------------
127.0.0.1 127.0.0.1 5 0 34235 lo0
-------------------------------------------------------
---------------------
value = 77 = 0x4d = 'M'
```

## rPing

Syntax:
**`rPing "source, "destination", "count"`**

Ping the destination from a remote source. Run this command from the VGMC> prompt.

The source parameter is the IP address or TN of the device requested to ping a destination IP address. The parameter destination is the IP address the source pings to. The count parameter is the number of successful attempts. If count is not specified, it is assigned a value of 4.

The following example shows command output on the VGMC.

```
VGMC> rPing "47.11.239.230", "47.11.213.216", "3"
value = 79 = 0x4f = 'O'

47.11.213.216 is the IP address of an IP set registered to
the SS.
47.11.239.230 is the IP address of the SS.
```

The following example shows command output on the Signaling Server.

```
PING 47.11.213.216 :  56 data bytes
64 bytes from 47.11.213.216:  icmp_seq=0.  time=0.  ms
64 bytes from 47.11.213.216:  icmp_seq=1.  time=0.  ms
64 bytes from 47.11.213.216:  icmp_seq=2.  time=0.  ms
64 bytes from 47.11.213.216:  icmp_seq=3.  time=0.  ms
----47.11.213.216 PING Statistics----
4 packets transmitted, 4 packets received, 0% packet loss
round-trip (ms) min/avg/max = 0/0/0
```

47.11.213.216 is the IP address of a Phase 2 IP Phone.

```
-> rPing "47.11.213.216", "47.11.239.230", "5"

value = 247379504 = 0xebeb630
```

```
->rPing Report from set (47.11.213.216):

64 bytes packets received from IP 47.11.239.230
ICMP sequence is 0
round trip time in ms:  0
rPing Report from set (47.11.213.216):

64 bytes packets received from IP 47.11.239.230
ICMP sequence is 1
round trip time in ms:  0

rPing Report from set (47.11.213.216):

64 bytes packets received from IP 47.11.239.230
ICMP sequence is 2
round trip time in ms:  0

rPing Report from set (47.11.213.216):

64 bytes packets received from IP 47.11.239.230
ICMP sequence is 3
round trip time in ms:  0

rPing Report from set (47.11.213.216):

64 bytes packets received from IP 47.11.239.230
ICMP sequence is 4
round trip time in ms:  0

rPing Report from set (47.11.213.216):

64 bytes packets received from IP 47.11.239.230
5 packets transimitted, 5 packets received, 0 packets lost
minimum round trip time in ms:  0
average round trip time in ms:  0
maximum round trip time in ms:  0
```

## rPingStop

Syntax:

**`rPingStop "source"`**

This command stops the rPing command from the source IP address or TN. Phase 2 IP Phones and the Softphone 2050 support this command. Run this command from the VGMC> prompt.

```
-> rPingStop "47.11.213.216"
value = 247379504 = 0xebeb630
```

## rtClockShow

Syntax:

**`rtClockShow`**

Display the date and time as known by the RTC chip. This command does not apply to the Signaling Server.

```
-> rtClockShow
RTC Time:  Date (12/02/2001) Time (16:42:29)
value = 44 = 0x2c = ','
```

## rTraceRoute

Syntax:

**`rTraceRoute "source", "destination", "max_hops"`**

Request one IP Phone to trace route destination IP address. Phase 2 IP Phones and the IP Softphone 2050 support this command. Run this command from the VGMC> prompt.

The parameter source is the IP address or TN of the IP Phone requested to ping a destination IP address. The parameter destination is the IP address the source pings to. The parameter max_hops is the maximum number of hops.

The following example is the output after the trace route report is received from the IP Phone.

```
rTraceRouteReport from Set (47.11.215.153)
1 -- 47.11.181.3 1.079ms 0.768ms 0.744ms
2 -- 47.11.174.10 2.861ms 2,654ms 2.690md
3 -- * * *
4 -- * * *
5 -- * * *
6 -- last packet

pdt> rTraceRoute "47.11.213.216", "47.11.239.230", "5"
pdt> 25/11/04 14:15:24 LOG0006 VTM: rTraceRoute Report from
set (47.11.213.216):
25/11/04 14:15:24 LOG0006 VTM: rTraceRoute Report from set
(47.11.213.216):
25/11/04 14:15:24 LOG0006 VTM: 1 -- 47.11.213.1:  0ms 30ms
10ms
25/11/04 14:15:24 LOG0006 VTM: rTraceRoute Report from set
(47.11.213.216):
25/11/04 14:15:24 LOG0006 VTM: 2 -- 47.11.248.221:  80ms
110ms 120ms
25/11/04 14:15:24 LOG0006 VTM: rTraceRoute Report from set
```

```
(47.11.213.216):
25/11/04 14:15:24 LOG0006 VTM: 3 -- 47.11.224.17:  10ms 0ms
0ms
25/11/04 14:15:24 LOG0006 VTM: rTraceRoute Report from set
(47.11.213.216):
25/11/04 14:15:24 LOG0006 VTM: 4 -- 47.11.239.230:  10ms
0ms 0ms
25/11/04 14:15:24 LOG0006 VTM: rTraceRoute completed !
```

## rTraceRouteStop

Syntax:

**`rTraceRouteStop "source"`**

Stop the previous trace route command through the VGMC. The IP Phone sends the response back to the VGMC. Phase 2 IP Phones and the IP Softphone 2050 support this command. Run this command from the VGMC> prompt.

```
pdt> rTraceRouteStop "47.11.213.216"
pdt>
```

## RTPStatShow

Syntax:

**`RTPStatShow "source"`**

Print the most recent copy of RTCP statistics report. Run this command from the vxWorksShell.

This command works regardless of whether the phone is active or inactive. If the phone is active and you run this command before the first RTCP statistics report is received, then the last RTCP statistics report from the previous call print. If the phone is not active and a prior call is made on this phone, the last RTCP statistics report from the previous call print. If no prior call is made on this phone before, all zero contents print.

The parameter source is the IP address or TN of the telephone being requested to show RTCP statistics.

## RTPTraceShow

Syntax:

**`RTPTraceShow "source", "number of polling period"`**

Use this command for expert support (vxWorksShell level). Use this command to obtain RTP/RTCP statistics for the specific IP end point with refreshing values in real time. Run this command from the vxWorksShell.

This command works regardless of whether the phone is active or inactive. If the phone is active, then the RTCP or RTCP-XR statistics prints for the specified period. If the phone is not active and a prior call is made on this phone, the last RTCP statistics report from the previous call print. If no prior call is made on this phone, all zero contents print.

The parameter source is the IP address of TN of the telephone being requested to show RTCP statistics. The parameter number of polling period is the number of times polling period RTCP statistics are displayed. If this parameter is not specified, the default is until the end of the call.

```
-> RTPTraceShow "47.11.213.216", "47.11.239.230", "5"
value = 0 = 0x0
```

## RTPTraceStop

Syntax:

**`RTPTraceStop "source"`**

Stop the previous command, RTPTraceShow. Phase 0/1 IP Phones, Phase 2 IP Phones, and IP Softphone 2050 support this command. Run this command from the vxWorksShell.

**`source`** is the IP address of TN of the telephone being requested to show RTCP statistics.

## rudpConfigShow

Syntax:

**`rudpConfigShow`**

This command displays the current configuration parameters for the RUDP connections.

```
-> rudpConfigShow Mode RUDP
Max retries 10
Timeout 400 ms
Bundle mode Off
Keep Alive 60000 ms
Payload ID 0xff
Window size 0
value = 21 = 0x15
```

## rudpShow

Syntax:

**`rudpShow`**

Show the status of all RUDP links on the VGMC/Signaling Server, including the link to the Call Server and all IP Phones registered with this card.

```
-> rudpShow
RUDP Port Summary
Port ID Src IP Src Port
+----------+-------------+--------+
0x039bff98 10.1.1.1 5100
0x039bf0e0 0.0.0.0 15001
0x039c0d88 192.168.1.201 15000
0x035a1d74 10.1.1.1 7300
0x03a88190 10.1.1.10 4100

RUDP Connection Summary

Src IP Src Port Connect ID Dst IP Dst Port Status Msg rcv Msg
sent Retries
+---------------+--------+----------+--------------+--
------+---------------+---------+---------+---------
-+
10.1.1.1 5100 0x03a88098 10.1.1.2 5000 ESTABLISHED <-> 44
1730 81
10.1.1.1 5100 0x035919d0 10.1.1.3 5000 ESTABLISHED <-> 13
1679 79
0.0.0.0 15001 0x039befe8 192.168.1.200 15000 DUDP 0 0 0
192.168.1.201 15000 0x039c0c90 192.168.1.200 15000
ESTABLISHED <-> 10820 538 1
value = 0 = 0x0
```

## RUDPStatShow

Syntax:

**`RUDPStatShow "source"`**

Display the information received from the IP endpoint, including number of messages sent, number of messages received and number of retries. Run this command from the VGMC> prompt.

The parameter source is the IP address or TN of the telephone being requested to provide the UNIStim/RUDP statistics.

The following example shows the RUDP/UNIStim statistic report output.

```
RUDP/UNIStim Statistic Report from Set (47.11.215.153)
Messages sent:  309
Messages received:  321
```

```
Number of retries:  10
Uptime of current TPS registration:  2 hour 24 minutes 35
seconds
```

> ***Note:*** Uptime is not cleared even if parameter clear is set in the command sent from Signaling Server/ VGMC to the telephone.

### RUDPStatShow

Syntax:

**`RUDPStatShow "source", "clear"`**

Clear the UNIStim/RUDP statistics counts when clear is assigned a value of 1. The uptime of Current TPS Registration is not cleared whether clear is assigned a value of 1 or not.

```
-> RUDPStatShow "47.11.213.216"
value = 247379504 = 0xebeb630
->

RUDPStatShow Report from set (47.11.213.216):
Number of Message Sent:  4878
Number of Message Received:  114977
Number of Retries:  4404
Number of Resets:  26
Uptime of Current TPS Registration:  0days 0hours 54minutes
34seconds
```

### serialNumShow

Syntax:

**`serialNumShow`**

Display the VGMC serial number data.

```
-> serialNumShow
Serial Number = NNTM1017213N 200023
```

Example output on Signaling Server:

```
-> serialNumShow
No serial number support.
value = 26 = 0x1a
```

### setClocks

Syntax:

**`setClocks yyyy, mm, dd, hh, mm, ss`**

Assign the time for the VGMC or MC32S realtime clock chip (RTC) and the VxWorks OS clock. The time and date is downloaded from OTM on Meridian 1 systems, or from the CS 1000. This command can be useful to remotely log on to a card on a Meridian 1 system. You must configure the time and date to a specific value be able to synchronize trace information. On CS 1000 systems, the time and data on the VGMC and Signaling Servers should already be synchronized with the CS 1000.

The VGMC application takes the time from the VxWorks OS clock. Normally that clock initializes from the RTC when the card boots up, when OTM down loads the time (on the Leader card), or when the SNTP response returns the time (on the Follower card). This command updates them both to the date and time entered by using the parameters yyyy, mm, dd, hh, mm, ss.

```
-> setClocks 2001,02,12,16,38,30
value = 0 = 0x0
```

## setLeader

Syntax:

**`setLeader "IPaddr","gwIPaddr","subnetMask"`**

Store all necessary information in the VGMC or MC32S card NVRAM so that on reboot the card starts up as a Leader 0. This command is not available on the Signaling Server. The parameter IPaddr is the card ELAN IP address, gwIPaddr is the ELAN gateway IP address and subnetMask is the ELAN subnet mask. This command sets the Leader flag and the BOOTP flag to retrieve the IP information from the NVRAM instead of sending a BOOTP request.

```
VGMC> setLeader "192.168.1.14","192.168.1.1","255.255.25
5.128"
IP address :  192.168.1.14
Gateway :  192.168.1.1
Subnet Mask:  255.255.255.128
Set as Leader.
Set to use IP address parameters in NVRAM.
Settings will take effect on card reboot.
value = 43 = 0x2b = '+'
```

## shellTimeoutDisable

Syntax:

**`shellTimeoutDisable`**

Keep the shell from timing out and allow unlimited access. This command is only available on the VGMC.

```
-> shellTimeoutDisable value = 0 = 0x0
```

### shellTimeoutEnable

Syntax:

**`shellTimeoutEnable`**

Restore the timeout function of the shell. The actual duration of the timeout can be assigned using the shellTimeoutSet command. This command is only available on the VGMC.

```
-> shellTimeoutEnable value = 0 = 0x0
```

### shellTimeoutGet

Syntax:

Display the currently configured idle shell timeout value in seconds. This command is only available on the VGMC.

```
-> shellTimeoutGet value = 1200 = 0x4b0
```

### shellTimeoutSet

Syntax:

**`shellTimeoutSet timeout`**

Assign the VGMC idle shell timeout value in seconds. This command is not available on the Signaling Server. The valid range for timeout is 30 to 4095. The default timeout is 1200 (20 minutes). The value is persistent: it is stored in NVRAM. The new timeout value takes effect immediately.

```
-> shellTimeoutSet 30 value = 0 = 0x0
```

### showMemConfig

Syntax:

**`showMemConfig`**

Display the size of the VGMC DRAM memory. This command is not available on the Signaling Server. Can be executed from the BIOS or VxWorks shells.

```
-> showMemConfig Memory Configuration:  Bank0:  64MB value
= 0 = 0x0
```

### sockShow

Syntax:

**`sockShow`**

Display a list of sockets in use.

The following example is from an ITG-P card.

```
-> sockShow
5 TCP AF_UNSPEC 0.0.0.0 :  21
6 TCP AF_UNSPEC 0.0.0.0 :  23
7 TCP AF_UNSPEC 0.0.0.0 :  1009
9 UDP AF_UNSPEC 0.0.0.0 :  69
10 UDP AF_UNSPEC 0.0.0.0 :  111
11 TCP AF_UNSPEC 0.0.0.0 :  111
12 UDP AF_UNSPEC 192.168.1.14 :  161
17 UDP AF_UNSPEC 0.0.0.0 :  67
19 UDP AF_UNSPEC 0.0.0.0 :  20000
21 UDP AF_UNSPEC 0.0.0.0 :  514
31 TCP AF_UNSPEC 0.0.0.0 :  1006
32 UDP AF_UNSPEC 0.0.0.0 :  15001
33 UDP AF_UNSPEC 192.168.1.14 :  15000
35 UDP AF_UNSPEC 0.0.0.0 :  16550
42 UDP AF_UNSPEC 192.168.1.140 :  7300
43 UDP AF_UNSPEC 0.0.0.0 :  16540
47 UDP AF_UNSPEC 192.168.1.149 :  4100
49 TCP AF_UNSPEC 192.168.1.14 :  23
51 UDP AF_UNSPEC 192.168.1.140 :  5100
value = 66775540 = 0x3fae9f4
->
```

The following example is from the Signaling Server.

```
-> sockShow
6 TCP AF_UNSPEC 0.0.0.0 :  21
7 UDP AF_UNSPEC 0.0.0.0 :  69
8 UDP AF_UNSPEC 0.0.0.0 :  111
9 TCP AF_UNSPEC 0.0.0.0 :  111
10 UDP AF_UNSPEC 10.11.216.166 :  161
15 UDP AF_UNSPEC 0.0.0.0 :  162
16 TCP AF_UNSPEC 0.0.0.0 :  513
17 TCP AF_UNSPEC 0.0.0.0 :  23
18 UDP AF_UNSPEC 0.0.0.0 :  67
21 UDP AF_UNSPEC 0.0.0.0 :  20031
23 TCP AF_UNSPEC 0.0.0.0 :  1009
24 UDP AF_UNSPEC 0.0.0.0 :  15001
25 UDP AF_UNSPEC 10.11.216.166 :  15000
26 UDP AF_UNSPEC 0.0.0.0 :  16550
28 UDP AF_UNSPEC 0.0.0.0 :  16501
29 UDP AF_UNSPEC 192.168.0.50 :  7300
30 UDP AF_UNSPEC 0.0.0.0 :  16540
```

```
32 UDP AF_UNSPEC 192.168.0.50 :  5100
35 UDP AF_UNSPEC 192.168.0.50 :  1719
36 UDP AF_UNSPEC 192.168.0.50 :  1718
37 TCP AF_UNSPEC 192.168.0.50 :  1720
38 UDP AF_UNSPEC 0.0.0.0 :  16500
39 TCP AF_UNSPEC 0.0.0.0 :  80
40 TCP AF_UNSPEC 10.11.216.166 :  1662
42 UDP AF_UNSPEC 192.168.0.51 :  1719
43 TCP AF_UNSPEC 192.168.0.51 :  1720
46 UDP AF_UNSPEC 192.168.0.51 :  4100
->
```

### SIPCallTrace on

Syntax:

**`SIPCallTrace on`**

Turn on tracing for all channels.

### SIPCallTrace off

Syntax:

**`SIPCallTrace off`**

Turn off tracing for all channels.

### SIPCallTrace ch channelNum MsgRecv MsgRecv

Syntax:

**`SIPCallTrace ch channelNum MsgRecv MsgRecv`**

Turn SIP tracing on or off.

The following table describes the command parameters.

**Table 44**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| channelNum | 0 to maximum channel number | Channel number of the virtual trunk to trace. |
| MsgRecv | ON<br>OFF | Specify whether to trace the messages sent to the specified channels. |
| MsgSend | ON<br>OFF | Specify whether to trace the messages sent from the specified channels. |

```
oam> SIPCallTrace ch 033 on on
oam> 11/01/05 15:22:11 LOG0006 SIPNPM: SIPCallTrace:
11/1/5 15:22:11 Recv chid:33 ip:192.168.19.50:5060 SIP
INVITE
11/01/05 15:22:11 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
```

```
15:22:11
Send chid:33 ip:192.168.19.51:5060 SIP response 100

11/01/05 15:22:11 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:11
Send chid:33 ip:192.168.19.51:5060 SIP response 180

11/01/05 15:22:11 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:11
Recv chid:33 ip:192.168.19.50:5060 SIP method PRACK(7)

11/01/05 15:22:11 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:11
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:22:19 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:19
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:22:19 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:19
Recv chid:33 ip:192.168.19.50:5060 SIP method ACK(1)

11/01/05 15:22:19 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:19
Recv chid:33 ip:192.168.19.50:5060 SIP method
other/unknown(6)

11/01/05 15:22:19 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:19
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:22:23 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:23
Send chid:33 ip:192.168.19.51:5060 SIP method BYE(2)

11/01/05 15:22:23 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:23
Recv chid:-1 ip:192.168.19.50:5060 SIP response 200

11/01/05 15:22:33 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:33
Recv chid:33 ip:192.168.19.50:5060 SIP INVITE

11/01/05 15:22:33 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:33
Send chid:33 ip:192.168.19.51:5060 SIP response 100
```

```
11/01/05 15:22:33 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:33
Send chid:33 ip:192.168.19.51:5060 SIP response 180

11/01/05 15:22:33 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:33
Recv chid:33 ip:192.168.19.50:5060 SIP method PRACK(7)

11/01/05 15:22:33 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:33
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:22:36 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:36
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:22:36 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:36
Recv chid:33 ip:192.168.19.50:5060 SIP method ACK(1)

11/01/05 15:22:36 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:36
Recv chid:33 ip:192.168.19.50:5060 SIP method
other/unknown(6)

11/01/05 15:22:36 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:22:36
Send chid:33 ip:192.168.19.51:5060 SIP response 200
```

### SIPCallTrace ch start_chNum end_chNum MsgRecv MsgSend

Syntax:

**`SIPCallTrace ch start_chNum end_chNum MsgRecv MsgSend`**

Enable tracing of a range of virtual trunk channels.

The following table describes the command parameters.

**Table 45**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| start_chNum | 0 to maximum channel number | Channel number to trace. |
| end_chNum | 0 to maximum channel number | Channel number to trace.<br><br>Must be greater than start_chNum. |

**Table 45**
**Command parameters (cont'd.)**

| Parameter | Value | Description |
|---|---|---|
| MsgRecv | ON<br>OFF | Specify whether to trace the messages sent to the specified channels. |
| MsgSend | ON<br>OFF | Specify whether to trace the messages sent from the specified channels. |

```
oam> SIPCallTrace ch 033 38 on on
oam> 11/01/05 15:23:30 LOG0006 SIPNPM: SIPCallTrace:
11/1/5 15:23:30 Send chid:33 ip:192.168.19.51:5060 SIP
method BYE(2)

11/01/05 15:23:30 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:23:30
Recv chid:-1 ip:192.168.19.50:5060 SIP response 200

11/01/05 15:23:40 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:23:40
Recv chid:33 ip:192.168.19.50:5060 SIP INVITE

11/01/05 15:23:40 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:23:40
Send chid:33 ip:192.168.19.51:5060 SIP response 100

11/01/05 15:23:40 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:23:40
Send chid:33 ip:192.168.19.51:5060 SIP response 180

11/01/05 15:23:40 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:23:40
Recv chid:33 ip:192.168.19.50:5060 SIP method PRACK(7)

11/01/05 15:23:40 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:23:40
Send chid:33 ip:192.168.19.51:5060 SIP response 200 oam>

11/01/05 15:24:00 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:24:0
Recv chid:33 ip:192.168.19.50:5060 SIP method CANCEL(8)

11/01/05 15:24:00 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:24:0
Send chid:33 ip:192.168.19.51:5060 SIP response 200
```

```
11/01/05 15:24:00 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:24:0
Send chid:33 ip:192.168.19.51:5060 SIP response 487
```

## SIPCallTrace num calling_number MsgRecv MsgSend

Syntax:

**`SIPCallTrace num calling/called_number MsgRecv MsgSend`**

Enable tracing of SIP messages using the called and calling numbers. If the called or calling number of a virtual trunk session matches the number specified, then the messages to and from the virtual trunk are traced.

The following table describes the command parameters.

**Table 46**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| calling_number | 1–32 | The telephone number to trace on. May be a partial calling or called number. |
| MsgRecv | ON<br>OFF | Specify whether to trace the messages sent to the specified channels. |
| MsgSend | ON<br>OFF | Specify whether to trace the messages sent from the specified channels. |

```
oam> SIPCallTrace 5500 on on
oam> 11/01/05 15:19:56 LOG0006 SIPNPM: SIPCallTrace:

11/1/5 15:19:56
Recv chid:33 ip:192.168.19.50:5060 SIP INVITE

11/01/05 15:19:56 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:56
Send chid:33 ip:192.168.19.51:5060 SIP response 100

11/01/05 15:19:56 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:56
Send chid:33 ip:192.168.19.51:5060 SIP response 180

11/01/05 15:19:56 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:56
Recv chid:33 ip:192.168.19.50:5060 SIP method PRACK(7)

11/01/05 15:19:56 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:56
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:20:00 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
```

```
15:20:0
Recv chid:33 ip:192.168.19.50:5060 SIP method CANCEL(8)

11/01/05 15:20:00 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:0
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:20:00 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:0
Send chid:33 ip:192.168.19.51:5060 SIP response 487

11/01/05 15:20:16 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:16
Recv chid:33 ip:192.168.19.50:5060 SIP INVITE

11/01/05 15:20:16 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:16
Send chid:33 ip:192.168.19.51:5060 SIP response 100

11/01/05 15:20:17 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:17
Send chid:33 ip:192.168.19.51:5060 SIP response 180

11/01/05 15:20:17 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:17
Recv chid:33 ip:192.168.19.50:5060 SIP method PRACK(7)

11/01/05 15:20:17 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:17
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:20:21 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:21
Recv chid:33 ip:192.168.19.50:5060 SIP method CANCEL(8)

11/01/05 15:20:21 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:21
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:20:21 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:21
Send chid:33 ip:192.168.19.51:5060 SIP response 487

11/01/05 15:20:26 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:26
Recv chid:33 ip:192.168.19.50:5060 SIP INVITE
```

```
11/01/05 15:20:26 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:26
Send chid:33 ip:192.168.19.51:5060 SIP response 100

11/01/05 15:20:26 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:26
Send chid:33 ip:192.168.19.51:5060 SIP response 180

11/01/05 15:20:26 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:26
Recv chid:33 ip:192.168.19.50:5060 SIP method PRACK(7)

11/01/05 15:20:26 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:26
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:20:30 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:30
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:20:30 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:30
Recv chid:33 ip:192.168.19.50:5060 SIP method ACK(1)

11/01/05 15:20:30 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:30
Recv chid:33 ip:192.168.19.50:5060 SIP method
other/unknown(6)

11/01/05 15:20:30 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:30
Send chid:33 ip:192.168.19.51:5060 SIP response 200

11/01/05 15:20:33 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:33
Recv chid:33 ip:192.168.19.50:5060 SIP method BYE(2)

11/01/05 15:20:33 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:20:33
Send chid:33 ip:192.168.19.51:5060 SIP response 200
```

## SIPCallTrace num calling_num NPI TON MsgRecv MsgSend

Syntax:

**`SIPCallTrace num calling_number NPI TON MsgRecv MsgSend`**

Trace SIP messages using the called and calling numbers. If the called or calling number of a virtual trunk session matches the number specified and the specified NPI and TON values match the call type, then the messages to and from the virtual trunk are traced.

The following table describes the command parameters.

**Table 47**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| calling_number | 1–32 | The telephone number to trace on. May be a partial calling or called number. |
| NPI | 0–7 | Specify the numbering plan identifier for which to trace calls.<br><br>0 = ALL NPIs<br>1 = Unknown<br>2 = ISDN/telephone numbering plan (E.164)<br>3 = Private numbering plan<br>4 = E.163<br>5 = Telex numbering plan<br>6 = Data numbering plan<br>7 = National standard numbering plan |
| TON | – | Specify the type of number to use as a filter for tracing. Only calls using this TON setting will be traced.<br><br>0 = All TONs<br>1 = Unknown Number<br>2 = International Number<br>3 = National Number<br>4 = Network Specific Number<br>5 = Subscriber Number<br>6 = L1 Regional Number<br>7 = L0 Regional Number |
| MsgRecv | ON<br>OFF | Specify whether to trace the messages sent to the specified channels. |
| MsgSend | ON<br>OFF | Specify whether to trace the messages sent from the specified channels. |

```
oam> SIPCallTrace num 5500 3 7 on on

The trace settings for Num:  5500, NPI: Private and TON: CDP
were already available as follows:
Number :  5500
NPI : Undefined
TON : Undefined
MsgRecv:  On
```

```
MsgSend:  On

oam> 11/01/05 15:19:19 LOG0006 SIPNPM: SIPCallTrace:
11/1/5 15:19:19 Recv chid:33 ip:192.168.19.50:5060 SIP
INVITE

11/01/05 15:19:19 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:19 Send chid:33 ip:192.168.19.51:5060 SIP response
100

11/01/05 15:19:19 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:19 Send chid:33 ip:192.168.19.51:5060 SIP response
180

11/01/05 15:19:19 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:19 Recv chid:33 ip:192.168.19.50:5060 SIP method
PRACK(7)

11/01/05 15:19:19 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:19 Send chid:33 ip:192.168.19.51:5060 SIP response
200

11/01/05 15:19:21 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:21 Send chid:33 ip:192.168.19.51:5060 SIP response
200

11/01/05 15:19:21 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:21 Recv chid:33 ip:192.168.19.50:5060 SIP method
ACK(1)

11/01/05 15:19:21 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:21 Recv chid:33 ip:192.168.19.50:5060 SIP method
other/unknown(6)

11/01/05 15:19:21 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:21 Send chid:33 ip:192.168.19.51:5060 SIP response
200

11/01/05 15:19:23 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:23 Recv chid:33 ip:192.168.19.50:5060 SIP method
BYE(2)

11/01/05 15:19:23 LOG0006 SIPNPM: SIPCallTrace:  11/1/5
15:19:23 Send chid:33 ip:192.168.19.51:5060 SIP response
200
```

### SIPGwShow

Syntax:

**`SIPGwShow`**

This command provides a snapshot summary of the state of the virtual trunk settings. This command assigns the channel ID a value of –1 by default. It does not show the virtual trunk status.

> ***Note:*** Channel ID must be a non-zero value.

```
oam> SIPGwShow
SIPNPM Status :  Active
Primary Proxy IP address :  192.168.19.51
Secondary Proxy IP address :  192.168.19.61
Primary Proxy port :  5060
Secondary Proxy port :  5060
Active Proxy :  Primary :Registered
Time To Next Registration :  2013 Seconds
Channels Busy / Idle / Total :  0 / 6 / 6
Stack version :  3.0.4.7
Channel tracing :  -1
Chan Direction CallState SIPState MediaState Codec AirTime
FS Fax DestNum RemoteIP
---- --------- --------- ---------------- -----------
---------------------- ------- --- --- -------
---------------
```

### SIPGwShow ch channelNum

Syntax:

**`SIPGwShow ch channelNum`**

Prints a snapshot summary of the state of the virtual trunk settings, plus the snapshot of the active call on the specified channel if the call exists.

The following table describes the command parameters.

**Table 48**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| channelNum | 0 to maximum channel number | Channel number to trace. |

```
oam> SIPGwShow ch 33
SIPNPM Status :  Active
Primary Proxy IP address :  192.168.19.51
Secondary Proxy IP address :  192.168.19.61
Primary Proxy port :  5060
```

```
Secondary Proxy port :  5060
Active Proxy :  Primary :Registered
Time To Next Registration :  1943 Seconds
Channels Busy / Idle / Total :  1 / 5 / 6
Stack version :  3.0.4.7
Channel tracing :  -1
Chan Direction CallState SIPState MediaState Codec AirTime
FS Fax DestNum RemoteIP
---- --------- --------- ----------------- -----------
---------------------- ------- --- --- -------
---------------
33 Terminate BUSY Invite Received SendRecv G_711_u_law_20M
S_NOVAD 61 Yes No 5801 192.168.19.155
```

## SIPGwShow num calling_number

Syntax:

**`SIPGwShow num calling_number`**

Print a snapshot summary of the state of the virtual trunk settings plus the snapshot of the active calls using the calling, called, or partial number specified.

The following table describes the command parameters.

**Table 49**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| calling_number | 0–32 | Telephone number to trace. |

```
oam> SIPGwShow num 5500
SIPNPM Status :  Active
Primary Proxy IP address :  192.168.19.51
Secondary Proxy IP address :  192.168.19.61
Primary Proxy port :  5060
Secondary Proxy port :  5060
Active Proxy :  Primary :Registered
Time To Next Registration :  1583 Seconds
Channels Busy / Idle / Total :  1 / 5 / 6
Stack version :  3.0.4.7
Channel tracing :  -1
Calling/Called Party Number:  5500
Numbering Plan Indicator:  Undefined
Type Of Number:  Undefined
Chan Direction CallState SIPState MediaState Codec AirTime
FS Fax DestNum RemoteIP
---- --------- --------- ----------------- -----------
---------------------- ------- --- --- -------
```

```
---------------
33 Terminate BUSY Invite Received SendRecv G_711_u_law_20M
S_NOVAD 10 Yes No 5801 192.168.19.155
```

## SIPGwShow num calling_number NPI TON

Syntax:

`SIPGwShow num calling_number NPI TON`

Prints a snapshot summary of the state of the virtual trunk settings, plus the snapshot of the active calls using the calling or called number or partial number with the specified NPI and TON values.

The following table describes the command parameters.

**Table 50**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| calling_number | 1–32 | The telephone number to trace on. May be a partial calling or called number. |
| NPI | 0–7 | Specify the numbering plan identifier for which to trace calls.<br><br>0 = ALL NPIs<br>1 = Unknown<br>2 = ISDN/telephone numbering plan (E.164)<br>3 = Private numbering plan<br>4 = E.163<br>5 = Telex numbering plan<br>6 = Data numbering plan<br>7 = National standard numbering plan |
| TON | 0–7 | Specify the type of number to use as a filter for tracing. Only calls using this TON setting are traced.<br><br>0 = All TONs<br>1 = Unknown Number<br>2 = International Number<br>3 = National Number<br>4 = Network Specific Number<br>5 = Subscriber Number<br>6 = L1 Regional Number<br>7 = L0 Regional Number |
| MsgRecv | ON<br>OFF | Specify whether to trace the messages sent to the specified channels. |
| MsgSend | ON<br>OFF | Specify whether to trace the messages sent from the specified channels. |

```
oam> SIPGwShow num 5500 3 7
SIPNPM Status :  Active
Primary Proxy IP address :  192.168.19.51
Secondary Proxy IP address :  192.168.19.61
Primary Proxy port :  5060
Secondary Proxy port :  5060
Active Proxy :  Primary :Registered
Time To Next Registration :  1524 Seconds
Channels Busy / Idle / Total :  1 / 5 / 6
Stack version :  3.0.4.7
Channel tracing :  -1
Calling/Called Party Number:  5500
Numbering Plan Indicator:  Private
Type Of Number:  CDP
Chan Direction CallState SIPState MediaState Codec AirTime
FS Fax DestNum RemoteIP
---- --------- --------- ----------------- -----------
---------------------- ------- --- --- -------
---------------
33 Terminate BUSY Invite Received SendRecv G_711_u_law_20M
S_NOVAD 69 Yes No 5801 192.168.19.155
```

### SIPOutput

Syntax:

**`SIPOutput output_destination "file_pathname"`**

Specify where to direct the output for the trace tool.

The following table describes the command parameters.

**Table 51**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| output_destination | 1–3 | Specify where to direct all trace messages for the SIPCallTrace.<br><br>1 = TTY<br>2 = RPTLOG<br>3 = file |
| file_pathname | "string" | Specify the file to output to, if output_destination = 3. Enclose the string in quotation marks. |

### SIPTraceShow

Syntax:

**`SIPTraceShow`**

Display trace settings, including the output destination, file name, and all active traces for the SIPCallTrace trace tool.

### spyHelp

Syntax:

**`spyHelp`**

Display the VxWorks help menu for the spy functions. The VxWorks spy function displays the task activity (real time usage) by monitoring the tasks and printing a summary at the specified interval. This command can help you determine if one task is running constantly and using all the CPU real time for the card. The monitoring and report printing itself uses some real time, so it is usually turned on, measurements taken, and then turned off.

```
-> spyHelp
spyHelp Print this list
spyClkStart [ticksPerSec] Start task activity monitor
running at ticksPerSec ticks per second
spyClkStop Stop collecting data
spyReport Prints display of task activity statistics
spyStop Stop collecting data and reports
spy [freq[,ticksPerSec]] Start spyClkStart and do a report
every freq seconds

ticksPerSec defaults to 100.  freq defaults to 5 seconds.
value = 0 = 0x0
```

### ssdShow

Syntax:

**`ssdShow`**

Displays the state of the SSD message trace.

```
-> ssdShow

Trace all input = off
Trace all output = off

Channel TN Handle I O
---------- ------ ---------- - -
0x3a873ac 0x6005 0x033cad78 0 0 0x33cd708 0x6006 0x033c9560
0 0
value = 33 = 0x21 = '!'
```

### ssdTrace

Syntax:

**`ssdTrace 0xSetTN, inMsg, outMsg`**

Controls the printing of SSD messages exchanged between the TPS and the Call Server CPU for a terminal. The default is off for both incoming and outgoing messages; the state is not saved and returns to off if the card is reset or reboots.

The following table describes the command parameters.

**Table 52**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| 0xSetTN | – | TN of the IP Phone, in hexadecimal format (preceded by "0x") To get telephone TN, use the tnsShow command. |
| inMsg | 0<br>1 | 0 = disables printing of SSD messages to the TPS<br>1 = enables printing of SSD messages to the TPS |
| outMsg | 0<br>1 | 0 = disables printing of SSD messages from the TPS card<br>1 = enables printing of SSD messages from the TPS card |

> ***Note:*** The enable all TN option (–1) on a busy TPS with caution due to the number of messages printed for each call. Remember, not only are call processing messages printed, and all lamp audit messages are printed.

You can temporarily turn off the lamp audit function from LD 77 (the password is 9950). Enter dlmp=1 to turn off lamp audit and dlmp=0 to turn it on. You can also turn on SSD messages at the Call Server side (for comparison) through LD 77. To turn on the messaging, enter `dmtn <tag #><TN>`. To turn off the messaging, enter `kill <tag #>`.

Entering only some of the parameters is the same as setting those not entered to zero, so `ssdTrace 0xTN` turns off printing in both directions, while `ssdTrace 0xTN,1` turns on printing for incoming messages and turn off outgoing message printing. A partial entry with "all TN" -1 provides a quick way to disable all printing: enter "ssdTrace -1".

Comparing the captured SSD trace with the one produced from LD 77 on the Call Server can identify if any SSD messages are lost. The point of reference for the direction of the messages is reversed when comparing the two log files: SSDO on the Call Server trace is SSDI on the ITG trace.

For each message, the following prints:

- task receiving or sending message
- timestamp when message was processed by the TPS
- syslog priority of this printout (all are Info messages)
- TN of the IP Phone sending or receiving the message

- direction of message flow relative to ITG (SSDI for incoming messages from the Call Server, or SSDO for outgoing messages to Call Server)
- a hexadecimal word of the SSD message contents

```
-> ssdTrace 0x6005,1,1
value = 56170928 = 0x35919b0
->

SEP 05 17:43:49 tSET: Info SSDO 6005 91FF
SEP 05 17:43:49 tSET: Info SSDI 6005 9404
SEP 05 17:43:49 tSET: Info SSDI 6005 9404
SEP 05 17:43:49 tSET: Info SSDI 6005 94C1
SEP 05 17:43:49 tSET: Info SSDI 6005 90F0
SEP 05 17:43:49 tSET: Info SSDI 6005 1401
SEP 05 17:43:51 tSET: Info SSDO 6005 91D4
SEP 05 17:43:51 tSET: Info SSDI 6005 1134
SEP 05 17:43:51 tSET: Info SSDO 6005 9114
SEP 05 17:43:52 tSET: Info SSDO 6005 91D0
SEP 05 17:43:52 tSET: Info SSDI 6005 1130
SEP 05 17:43:52 tSET: Info SSDO 6005 9110
SEP 05 17:43:52 tSET: Info SSDO 6005 91D0
SEP 05 17:43:52 tSET: Info SSDI 6005 1130
SEP 05 17:43:52 tSET: Info SSDO 6005 9110
SEP 05 17:43:52 tSET: Info SSDO 6005 91D0
SEP 05 17:43:52 tSET: Info SSDI 6005 1130
SEP 05 17:43:52 tSET: Info SSDI 6005 1406
SEP 05 17:43:52 tSET: Info SSDO 6005 9110
SEP 05 17:43:54 tSET: Info SSDI 6005 1408
SEP 05 17:43:58 tSET: Info SSDO 6005 913F
SEP 05 17:43:58 tSET: Info SSDI 6005 9401
SEP 05 17:43:58 tSET: Info SSDI 6005 9404
SEP 05 17:43:58 tSET: Info SSDI 6005 1400
SEP 05 17:43:58 tSET: Info SSDI 6005 1400
SEP 05 17:43:58 tSET: Info SSDI 6005 9000
```

## syslogLevelSet

Syntax:

**`syslogLevelSet task, level`**

Control the printing of detailed information from a task. To change information for multiple tasks, enter this command multiple times.

The following table describes the command parameters.

**Table 53**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| task | – | Specify the module for this entry of the command. The name of the task as printed by the logShow (VGMC), syslogShow or i (VGMC/Signaling Server) command is used.<br>tSnmpd = SNMP daemon task<br>tSntpsTask = SNTP Server task<br>tXA, tA07, tRDP = SSD driver interface<br>tMAM = Maintenance Admin Module<br>tMVX_XSPY = Telogy SPY (debugger)<br>tMVX_DIM = Telogy Dsp Interface Module<br>tOMM = Operational Measurement Module<br>tVTM = Virtual Terminal Manager<br>tVTI = Virtual Terminal Interface<br>tSET = Digital Set Emulator<br>tCSV =Connect Server<br>tTPS = Terminal Proxy Server<br>tTCK = TPS Socket Task<br>tUMS = UMS Server<br>tUMC = UMS Client<br>tVGW = Voice Gateway<br>tRTP = Real Time Protocol<br>tRTCP = Real Time Control Protocol<br>tTelnetOutTask = Telnet output task<br>tTelnetInTask = Telnet input task tyLstnr<br>tShell = VxWorks Shell task |
| level | 0–7 | Determines the information printed. Assigning a level to a task ensures messages of that level and lower to print.<br><br>0 = Emergency, system is unusable<br>1 = Alert, action must be taken immediately<br>2 = Critical, critical conditions<br>3 = Error, error conditions<br>4 = Warning, warning conditions<br>5 = Notice, normal but significant condition<br>6 = Info, informational<br>7 = Debug, debug level messages<br><br>By default, most tasks are set to Info. When troubleshooting a problem, it is often useful to set the level to Debug to get more detailed information from a task. |

```
-> syslogLevelSet tMAM, 7
value = 6 = 0x6
```

## syslogShow

Syntax:

**`syslogShow`**

On the Signaling Server, logShow provides little information, so use this command to obtain the log output level for the task. On the VGMC, this command is similar to the logShow command, but prints only the task and level information and works only from the VxWorks shell. Use this command with the syslogLevelSet command.

The following example shows the Signaling Server output (includes not only LTPS tasks but also VTRK and Gatekeeper).

```
-> syslogShow
Task Level
-------------------- -------
tSysWork none
tExcTask none
tLogTask none
tAioWait none
tAioIoTask1 none
tAioIoTask0 none
tNetTask none
tPortmapd none
tFtpdTask none
tTftpdTask none
tWdbTask none
tRptd none
tLogin none
tLogin none
tRLogind none
tTelnetd none
tSnmpd none
tPxTimer none
tbootpd none
tSNTPC Info
tRDP Info
tMAM Info
tOMM Info
tELC Info
tVTM Info
tVTI Info
tSET Info
tCSV Info
```

```
tTPSAR Info
tTPS Info
tTPSARReceive none
tfwBk Info
tUMS Info
tUMC Info
tVTK Info
tNPM Info
tXMSG Info
tHTTPd none
tHTTPd none
tHTTPd none
tHTTPd none
shell none
tPBX Info
tTelnets38 none
tLogin none
tTelnetc38 none
shell none
tShell Info
value = 5 = 0x5
```

The following example shows command output for ITG-P.

```
-> syslogShow
Task Level
-------------------- -------
tExcTask none
tLogTask Info
tAioWait none
tAioIoTask1 none
tAioIoTask0 none
tPcmciad none
tTffsPTask none
tNetTask Info
tTelnetd none
tPortmapd none
tRdbTask none
tFtpdTask none
tTftpdTask none
tSnmpd none
tSyslogd Info
tMonTask none
tPxTimer none
tbootpd Info
tSNTPC Info
baseMMintTask none
```

```
tXA Info
tA07 Info
tRDP Info
tMAM Info
tMVX_XSPY Info
tMVX_DIM Info
tOMM Info
tRPCMGMT none
tELC Info
tVTM Info
tVTI Info
tSET Info
tCSV Info
tTPS Info
tfwBk Info
tUMS Info
tUMC Info
tVGW Info
tRTP Info
tRTCP Info
midnightTask none
tTelnetOutTask none
tTelnetInTask none
tyLstnr none
tShell Info
value = 5 = 0x5
```

The following example shows command output for SMC.

```
-> syslogShow
Task Level
-------------------- -------
tExcTask none
tLogTask none
tAioWait none
tAioIoTask1 none
tAioIoTask0 none
tPcmciad none
tDcacheUpd none
tNetTask none
tTelnetd none
tPortmapd none
tFtpdTask none
tTftpdTask none
tSnmpd none
tSyslogd none
tMonTask none
```

```
tPxTimer none
tbootpd none
tSNTPC none
baseMMintTask none
tXA Info
tA07 Info
tRDP Info
tMAM Info
tMVX_XSPY none
tMVX_DIM Info
tOMM Info
tPBX Info
tELC Info
tVGW Info
tRTP Info
tRTCP Info
tXMSG Info
midnightTask none
tShell none
tTelnetOutTask none
tTelnetInTask none
tyLstnr none
value = 5 = 0x5
```

## swDownload

Syntax:

**`swDownload "srvrIP", "uid", "passwd", "path", "fname"`**

Download the VGMC application binary from the specified FTP server, compress it and store it in the VGMC flash memory. This command applies only to the VGMCs. Run the command from the VGMC, VxWorks and BIOS CLIs.

> ***Note:*** The VGMC application binary can exceed 90 percent of disk space on the C: drive of the ITG-P card, which results in a G012(ITG1012) error message. Nortel recommends that you delete the application binary from the C: drive after you upgrade the card.

The following table describes the command parameters.

**Table 54**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| srvrIP | – | IP address of the FTP server.<br>If the FTP server resides on the VGMC itself, enter the loopback address 127.0.0.1 for srvrIP. This directs the command to the card on which the command is issued. |
| uid | – | Server logon user ID.<br>For the Signaling Server or VGMCs, the user ID is the CS 1000 PWD1 logon after the card connects to the Call Server. For new cards, the user ID is itgadmin. |
| passwd | – | Server logon password.<br>For the Signaling Server/VGMCs, the password is the CS 1000 PWD1 logon after the card connects to the Call Server. For new cards, the password is itgadmin. |
| path | – | Path on the server of the file to retrieve.<br>Drive letters must be capitalized, for example, C:/path. |
| fname | – | File name of the file to retrieve.<br>Must be in 8.3 format. |

The following example is the output for a file located on the Call Server.

```
VGMC> swDownload "Call Server IP address","user
ID","password","path","filename"
```

## swVersionShow

Syntax:
**`swVersionShow`**

Print the card VGMC application software version.

The following example shows command output on the ITG-P card.

```
VGMC> swVersionShow
Installed Image:  VGMC SSE-2.00.70_VGMC-3.00.70_10_07_200
2.2099 (ITGPentium) -
Monday October 7 13:38:40 EDT 2002
value = 0 = 0x0
```

The following example shows command output on the SMC (same as ITG-P output).

```
-> swVersionShow
Installed Image:  VGMC SSE-2.00.70_VGMC-3.00.70_10_07_200
2.2099 (ITGSA) -
Monday October 7 13:44:38 EDT 2002
value = 0 = 0x0
```

The following example shows command output on the Signaling Server.

```
-> swVersionShow
sse-2.00.74 Wednesday October 16 2002 20:04:18 EDT
Loaded Modules:
share.obj sse-2.00.74
line.obj sse-2.00.74
trunk.obj sse-2.00.74
gk.obj sse-2.00.74
web.obj sse-2.00.74
value = 0 = 0x0
```

### tcpstatShow

Syntax:

**`tcpstatShow`**

Display the TCP protocol statistics.

```
-> tcpstatShow
TCP:
0 packet sent
0 data packet (0 byte)
0 data packet (0 byte) retransmitted
0 ack-only packet (0 delayed)
0 URG only packet
0 window probe packet 0 window update packet
0 control packet
0 packet received
0 ack (for 0 byte)
0 duplicate ack
0 ack for unsent data
0 packet (0 byte) received in-sequence
0 completely duplicate packet (0 byte)
0 packet with some dup.  data (0 byte duped)
0 out-of-order packet (0 byte)
0 packet (0 byte) of data after window
0 window probe
0 window update packet
0 packet received after close
0 discarded for bad checksum
0 discarded for bad header offset field
```

```
0 discarded because packet too short
0 connection request
0 connection accept
0 connection established (including accepts)
0 connection closed (including 0 drop)
0 embryonic connection dropped
0 segment updated rtt (of 0 attempt)
0 retransmit timeout
0 connection dropped by rexmit timeout
0 persist timeout
0 keepalive timeout
0 keepalive probe sent
0 connection dropped by keepalive
0 pcb cache lookup failed
value = 27 = 0x1b
```

## tLanDuplexSet

Syntax:

**`tLanDuplexSet duplexMode`**

Configure the duplex mode of the TLAN interface while operating in 10BaseT mode. The command takes effect immediately. The duplex mode setting is saved in NVRAM and read at startup.

The following table describes the command parameters.

**Table 55**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| duplexMode | 0<br>1 | 0 = enables full duplex mode<br>1 = enables half duplex mode |

## tLanSpeedSet

Syntax:

**`tLanSpeedSet`**

Configure the speed of the TLAN interface. By default, this interface autonegotiates to the highest speed supported by the hub or switch; with a 10/100BaseT switch, the interface negotiates to 100BaseT. This command is useful to debug Ethernet speed-related problems and to force the interface to 10BaseT operation; the command takes effect immediately. The duplex mode setting is saved in NVRAM and read at startup.

Use the command CardShow to check the current speed of the TLAN interface.

The following table describes the command parameters.

**Table 56**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| speed | 10<br>10100 | 10 = enable 10MB-only operation<br>10100 = enable autonegotiation |

### tpsARShow

Syntax:
**`tpsARShow`**

Display information about the Gatekeeper address resolution task (tpsAR).

```
-> tpsARShow
Active Connect Server Type :  Primary
Active Connect Server IP : 192.168.0.50
Active Connect Server port :  16500
Currently there are 0 outstanding DN to IP translation
requests
2 Connect Servers Configured
========================================
Primary Connect Server IP : 192.168.0.50
Primary Connect Server Port :  16500
Alternate Connect Server IP : 23.24.25.26
Alternate Connect Server Port :  16500
Connect Server Timeout :  10
========================================
value = 41 = 0x29 = ')'
```

### tpsARTrace

Syntax:
**`tpsARTrace "type", "trace_id"`**

Turn on the trace for a specified identifier, trace_id. It allows the tracing of the tpsAR protocol used to determine where a telephone registers to.

The following table describes the command parameters.

**Table 57**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| type | ALL<br>IP<br>ID | Trace type.<br><br>ALL = any trace type<br>IP = search on an IP address<br>ID = search on an identifier, such as DN or H323 alias |
| trace_id | "string" | Trace identifier, such as an IP address, DN, or alias. |

The following example shows a search on an NCS, node, or TLAN IP address.

```
tpsARTrace "IP","192.168.2.2"
```

The following example shows a search on an H323 alias.

```
tpsARTrace "ID","MyH323Alias"
```

The following example shows a search on a DN that is logging in.

```
tpsARTrace "ID","7778"
```

### tpsARTraceHelp

Syntax:
**`tpsARTraceHelp`**

Print a list of all CLIs for tracing the tpsAR protocol messages. The output describes each CLI parameter.

### tpsARTraceSettings

Syntax:
**`tpsARTraceSettings`**

Display the current trace settings.

### tpsAROutput

Syntax:
**`tpsAROutput trace_output, "file_name"`**

Assign the output destination for the trace tool.

The following table describes the command parameters.

**Table 58**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| trace_output | 1–4 | An integer value specifying the trace output destination:<br>1 = TTY<br>2 = SYSLOG<br>3 = File<br>4 = File and TTY |
| filename | "string" | Specify the file to output to if trace_output = 3 or 4.<br><br>You can enter the complete path name. If you enter only the file name, the default path is C:/file_name.<br><br>Enclose the string in quotation marks. |

```
-> tpsAROutput 1, "trace.txt"
value = 0 = 0x0
```

### tpsARTraceOff

Syntax:

**`tpsARTraceOff "trace_id"`**

Turn off the trace for the specified trace identifier, trace_id.

```
-> tpsARTraceOff "7778"
value = 0 = 0x0
```

### tpsARTraceAllOff

Syntax:

**`tpsARTraceAllOff`**

Turn off the trace for all trace identifiers.

### tpsRemoteShow

Syntax:

**`tpsRemoteShow level`**

Display the current Gate Keeper query or Remote Monitoring query list under monitoring by TPS task for Virtual Office and Branch Office feature. The parameter level controls the amount of displayed information. The default value of 0 shows the basic GK query and RM query list. Values greater than 0 shows additional details about the GK and RM query.

```
-> tpsRemoteShow

TPS Remote Communication Manager
```

```
********* GK Query List ***********
Outstanding Query:  0
TimeoutUnit:  500 ms
GK query timeout:  300
GK query timeout count:  0

********* Remote TPS monitoring ***********
Outstanding Monitors:  0
TimeoutUnit:  500 ms
Num of Polls:  10
Monitor Interval 150000 ms
Next timeout in 250 msec
value = 25 = 0x19
->

-> tpsRemoteShow 4
TPS Remote Communication Manager
********* GK Query List ***********
Outstanding Query:  1
TimeoutUnit:  500 ms
GK query timeout:  300
GK query timeout count:  22
requestNumber = 0xffff, ton = 4, DN = 2011, retry = 0xfefe,
retryType = 0x100

********* Remote TPS monitoring ***********
Outstanding Monitors:  0
TimeoutUnit:  500 ms
Num of Polls:  10
Monitor Interval 150000 ms
Next timeout in 400 msec value = 0 = 0x0
```

## tpsShow

Syntax:

**`tpsShow`**

List information relevant to the TPS. If more than one card in the system, then this table (on the Leader card only) lists all VGMC or Signaling Servers in the node with their IP addresses and TN. If a card does not appear that should, then the registration process between the Follower and Leader was not successful.

The following table describes the data parameters output by this command and how to interpret them.

**Table 59**
**Data output**

| Parameter | Description |
|---|---|
| Node ID | Node this card belongs to |
| Is master | Set to 1 if this card is the master |
| Up Time | Duration in seconds the card has been up |
| TN | Card physical TN (not displayed for Signaling Server) |
| IP TLAN | IP address of card on the TLAN |
| IP ELAN | IP address of card on the ELAN |
| ELAN Link | Status of ELAN link with Call Server (not port carrier status) |
| Sets Connected | Number of telephones registered with this TPS card |
| Sets Reserved | Sets with a spot reserved on this card but have not yet completed reservation process |

The following example shows command output on the SMC.

```
VGMC> tpsShow
Node ID : 3556
Is master :  0
Up time :  2 days, 22 hours, 23 mins, 35 secs (253415 secs)
TN : 05-00
Platform :  ITG SA
TPS Service :  Yes
IP TLAN : 47.11.215.143
IP ELAN : 47.11.216.174
ELAN Link :  Up
Sets Connected:  2 Sets
Reserved :  0
```

Example Signaling Server output:

```
-> tpsShow
Node ID : 555
Is master :  1
Up time :  1 days, 22 hours, 46 mins, 11 secs (168371 secs)
Platform :  ISP 1100
TPS Service :  Yes
IP TLAN : 47.11.249.96
IP ELAN : 47.11.255.26
ELAN Link :  Up
Sets Connected:  5
Sets Reserved :  0
```

## tpsSocketShow

Syntax:

**`tpsSocketShow`**

Display information about the TPS sockets used for communication with other cards.

The following example shows output from the Master node.

```
-> tpsSocketShow Communication Style:  Broadcast
Tx port :  16543
Tx Socket :  40
Tx socket addr :  192.168.1.255
Rx port :  16543
Rx IP : 0.0.0.0
Rx Socket :  40
Rx socket addr :  0.0.0.0
value = 29 = 0x1d
```

The following example shows output from a Follower node.

```
-> tpsSocketShow
Communication Style:  Multicast
Tx port :  16543
Tx Socket :  41
Tx socket addr :  192.168.1.140
Rx port :  16543
Rx IP : 0.0.0.0
Rx Socket :  41
Rx socket addr :  0.0.0.0
value = 29 = 0x1d
```

## tpsSocketTraceSet

Syntax:

**`tpsSocketTraceSet traceState`**

This message trace is useful for debugging Master election problems.

The following table describes the command parameters.

**Table 60**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| traceState | 0<br>1 | 1 = enable printing of a message each time a message is received or transmitted from the TPS task<br>0 = disable the message printing (default) |

The following example shows output from the Master node.

```
-> tpsSocketTraceSet 1
value = 1 = 0x1
->
FEB 23 12:36:49 tTPS: Info change comm style to Broadcast
FEB 23 12:36:49 tTCK: Info socket msg from 192.168.1.140,
len 16
FEB 23 12:36:49 tTCK: Info socket msg from 192.168.1.140,
len 16
FEB 23 12:36:49 tTPS: Info send msg from 192.168.1.140 to
192.168.1.255
```

### tsm_set_rx_gain chNum, gain_value

Syntax:

**`tsm_set_rx_gain chNum, gain_value`**

Assign a DSP channel receive gain (gain for audio in from the IP Phone) for the active call. This command applies only to the VGMCs.

After the call is dropped, the gain adjustment is reset. This command is intended for temporary problem debugging only, not for actual loss plan changes.

The following table describes the command parameters.

**Table 61**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number. |
| gain_value | –14 to 14 | DSP gain, in the range of –14 to +14 dB.<br>Positive values are louder. |

```
-> tsm_set_rx_gain 0,6
value = 0 = 0x0
```

### tsm_set_tx_gain chNum, txGain, txInGain

Syntax:

**`tsm_set_tx_gain chNum, txGain, txInGain`**

This command sets a DSP channel transmit gain (gain for audio out towards the IP Phone) for the active call. This command applies only to the VGMCs.

The following table describes the command parameters.

**Table 62**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number of the call on the VGMC. |
| txInGain | –14 to 14 | Gain register located between the TDM interface and the echo canceller. Assigns the DSP gain, in the range of –14 to +14 dB. Positive values are louder. |
| txGain | –14 to 14 | Gain register located between the echo canceller and the IP network. Assigns the DSP gain, in the range of –14 to +14 dB. Positive values are louder.<br><br>This parameter provides more control over the audio levels of the echo canceller. |

```
-> tsm_set_tx_gain 0,-4, 6
value = 0 = 0x0
```

## tsm_stat_req_ecdbg

Syntax:

**`tsm_stat_req_ecdbg chNum`**

Return the echo canceller debug statistics for a channel. This command applies only to the VGMCs.

The following table describes the command parameters.

**Table 63**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number of the call on the VGMC. |

The following table describes the data parameters printed by this command.

**Table 64**
**Data output**

| Parameter | Description |
|---|---|
| Px level<br>Py level<br>Pe level | Instantaneous power level of far end (IP Phone talker), near end (remote) user, and error level (echo residual before NLP). Max value is 228 (268,435,456) = +3 dbm0 (All values are in Q4). |
| erl_estimate | Estimated ERL in Q4 (hybrid loss); dividing by 16 gives ERL in dB. |
| acom_estimate | Estimated ACOM in Q4 |

**Table 64**
**Data output (cont'd.)**

| Parameter | Description |
|---|---|
| len | length [x] of internal use array |
| internal_use[0,x] | Values of internal use array |

The following example shows the output for an application load with the version 7.0.6.38 Telogy code.

```
MAR 18 15:40:03 tMVX_DIM: Info TSG: 0 Echo Canceller Debug
Stats
MAR 18 15:40:03 tMVX_DIM: Info Px level = -739
MAR 18 15:40:03 tMVX_DIM: Info Py level = -962
MAR 18 15:40:03 tMVX_DIM: Info Pe level = -963
MAR 18 15:40:03 tMVX_DIM: Info acom_estimate = 511
MAR 18 15:40:03 tMVX_DIM: Info len = 25
MAR 18 15:40:03 tMVX_DIM: Info ERLE estimate = 521
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[0] = 0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[1] =
0x150
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[2] = 0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[3] =
0x110
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[4] = 0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[5] = 0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[6] = 0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[7] = 0x7
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[8] = 0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[9] =
0x1c
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[10] =
0x1
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[11] =
0x3f33
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[12] =
0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[13] =
0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[14] =
0x2
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[15] =
0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[16] =
0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[17] =
0xa
```

```
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[18] =
0x7
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[19] =
0xf24c
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[20] =
0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[21] =
0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[22] =
0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[23] =
0x0
MAR 18 15:40:03 tMVX_DIM: Info TCID 0, internal_use[24] =
0x0
```

### tsm_stat_req_magdbg

Syntax:

**`tsm_stat_req_magdbg chNum, clear`**

Return the echo canceller MIPS agent debug statistics for a given channel. This command applies only to the VGMCs.

Do not use this command on the ITG-P unless possible card reset is not a problem.

The following table describes the command parameters.

**Table 65**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number of the call on the VGMC. |
| clear | 0<br>1 | 0 = Do not clear statistics after display.<br>1 = Clear statistics after display. |

```
-> tsm_stat_req_magdbg 0,0
value = 0 = 0x0
->
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[0] = 0x1
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[1] =
0x7c00
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[2] = 0x0
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[3] =
0x11cd
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[4] = 0x0
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[5] = 0x0
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[6] = 0x1
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[7] =
0x6a33
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[8] = 0x1
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[9] =
0x5c3f
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[10] =
0x0
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[11] =
0x9a4
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[12] =
0x0
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[13] =
0x22ce
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[14] =
0x0
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[15] =
0x1446
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[16] =
0x1
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[17] =
0x1
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[18] =
0xe24c
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[19] =
0x400
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[20] =
0x100
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[21] =
0x100
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[22] =
0x1
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[23] =
0x1
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[24] =
0x450
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[25] =
0x11cd
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[26] =
0x0
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[27] =
0x0
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[28] =
0x22ce
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[29] =
0x1446
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[30] =
0x9a4
MAR 19 09:30:07 tMVX_DIM: Info TCID 0, internal_use[31] =
0x0
```

### tsm_stat_req_error

Syntax:

**`tsm_stat_req_error chNum`**

Return the current errors detected by the DSP for the specified channel. This command applies only to the VGMCs. The variable DimDspStat = must be assigned a value of 1 for this command to produce output.

The following data appears:

- Number of incoming voice packets dropped by the DSP due to invalid generic voice header syntax.
- Number of transmit voice packets dropped due to voice buffer overflow (buffer to micro was busy when DSP attempted to send packet).
- Estimated number of lost incoming enhancement voice packets (two core packets received without intervening enhancement). This is only for E-ADPCM, which is not used on this product.
- Number of dropped incoming enhancement voice packets due to absence of a core packet. This is only for E-ADPCM, which is not used on the CS 1000.
- Number of packets lost by the network, that is, from missing incoming RTP sequence numbers.

You can enter this command repeatedly during a call. The counter values are not reset until a new call is made.

The following table describes the command parameters.

**Table 66**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number of the call on the VGMC. |

The following example shows command output for ITG-P.

```
-> tsm_stat_req_error 0
value = 0 = 0x0
->
JAN 01 14:33:55 tMVX_DIM: Info TCID 0, invalid_header_count
= 0
JAN 01 14:33:55 tMVX_DIM: Info TCID 0, to_micro_overflow_co
unt = 0
```

```
JAN 01 14:33:55 tMVX_DIM: Info TCID 0, lost_enh_packet_coun
t = 0
JAN 01 14:33:55 tMVX_DIM: Info TCID 0, no_core_packet_count
= 0
JAN 01 14:33:55 tMVX_DIM: Info TCID 0, pkt_lost_by_network
= 0 ->
```

Example SMC output:

```
-> tsm_stat_req_error 0
value = 0 = 0x0
->
NOV 21 15:19:52 tMVX_DIM: Info TCID 0, invalid_header_count
= 0
NOV 21 15:19:52 tMVX_DIM: Info TCID 0, to_micro_overflow_co
unt = 0
NOV 21 15:19:52 tMVX_DIM: Info TCID 0, rx_routing_dropped =
0
NOV 21 15:19:52 tMVX_DIM: Info TCID 0, lost_enh_packet_coun
t = 0
NOV 21 15:19:52 tMVX_DIM: Info TCID 0, no_core_packet_count
= 0
NOV 21 15:19:52 tMVX_DIM: Info TCID 0, pkt_lost_by_network
= 0
NOV 21 15:19:52 tMVX_DIM: Info TCID 0, rc4key_update_lost_p
kt_count = 0
NOV 21 15:19:52 tMVX_DIM: Info TCID 0, invalid_mac_header_c
ount = 0
NOV 21 15:19:52 tMVX_DIM: Info TCID 0, invalid_ssrc_count
= 0 NOV 21 15:19:52 tMVX_DIM: Info TCID 0, invalid_payload
_count = 0 ->
```

## tsm_stat_req_rx_tx

Syntax:

**`tsm_stat_req_rx_tx chNum`**

Return data for the current call on the channel. This command applies only to the VGMCs. You must assign the variable DimDspStat = a value of 1 for this command to produce output.

You can repeat the query for the duration of the call. The counts are reset at the beginning of the next call.

Receive (RX) and Transmit (TX) are from the card perspective. Receive (RX) is from the telephone, and Transmit (TX) is to the telephone.

This command produces the following output:

- number of Receive (RX) packets sent for playout
- number of Transmit (TX) packets the DSP has written to the voice data buffer
- number of voice frames on Transmit (TX) side classified as silence
- minimum jitter
- maximum jitter

When VAD is disabled, the Receive (RX) and Transmit (TX) packet counts are close in value.

The following table describes the command parameters.

**Table 67**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number of the call on the VGMC. |

Example ITG-P or SMC card output:

```
-> tsm_stat_req_rx_tx 0
value = 0 = 0x0
->
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, rx_packet_count =
2979
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, tx_packet_count =
2983
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, silence_packet_count
= 0
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, min_jitter = 18
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, max_jitter = 22
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, rtp_average_jitter =
0
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, tx_grant_sync_drop =
0
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, tx_octets = 477280
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, rx_octets = 476640
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, aal2_cod_prof_chgs =
0
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, dtmf_tx_octets = 0
JAN 01 14:33:39 tMVX_DIM: Info TCID 0, dtmf_rx_octets = 0
```

## tsm_stat_req_tele_levels

Syntax:

**`tsm_stat_req_tele_levels chNum`**

Return the current audio levels measured in the DSP for the specified gateway channel. This command applies only to the VGMCs. Receive (RX) and Transmit (TX) are from the VGMC perspective. Transmit (TX) means towards the TDM and Receive (RX) means from it. The following data is displayed:

- Receive (RX) power in 0.1 dBm units
- Transmit (TX) power in 0.1 dBm units
- Receive (RX) mean in 0.1 linear PCM units (ignore, apparently is not accurate)
- Transmit (TX) mean in 0.1 linear PCM units (ignore, apparently is not accurate)

You can repeat the query for the duration of the call. The counts are reset at the beginning of the next call.

The following table describes the command parameters.

**Table 68**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number of the call on the VGMC. |

```
-> tsm_stat_req_tele_levels 0
value = 0 = 0x0
->
JAN 17 14:44:59 tMVX_DIM: Info TCID =0 rx_level = -610
JAN 17 14:44:59 tMVX_DIM: Info TCID =0 tx_level = -610
JAN 17 14:44:59 tMVX_DIM: Info TCID =0 rx_mean = -40
JAN 17 14:44:59 tMVX_DIM: Info TCID =0 tx_mean = -20 ->
```

## tsm_stat_req_vp_delay

Syntax:

**`tsm_stat_req_vp_delay chNum`**

Return the DSP Voice Playout Unit (VPU) statistics for the call on the channel specified. This command applies only to the VGMCs. The variable DimDspStat = must be assigned a value of 1 for this command to produce output. The VPU operates on segments of 10 milliseconds of voice. The following data prints:

- number of lost segments during the call based on the missing sequence number
- number of segments replayed due to the lost packets
- number of idle (noise) segments being played out due to either lost packets or voice playout FIFO buffer underrun

- number of voice segments dropped due to the voice playout FIFO buffer over-run
- total number of segments submitted by packetization unit for playout
- average frame jitter seen from network (in Telogy 7.01 output only).

You can repeat the query for the duration of the call. The counts are reset at the beginning of the next call.

The following table describes the command parameters.

**Table 69**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number of the call on the VGMC. |

```
-> tsm_stat_req_vp_delay 0
value = 0 = 0x0
->
JAN 01 14:33:21 tMVX_DIM: Info TCID 0, avg_playout_delay =
60
JAN 01 14:33:21 tMVX_DIM: Info TCID 0, lost_packet_count = 0
JAN 01 14:33:21 tMVX_DIM: Info TCID 0, replay_packet_count
= 0
JAN 01 14:33:21 tMVX_DIM: Info TCID 0, idle_packet_count = 0
JAN 01 14:33:21 tMVX_DIM: Info TCID 0, dropped_packet_count
= 0
JAN 01 14:33:21 tMVX_DIM: Info TCID 0, rx_packet_count =
4140
JAN 01 14:33:21 tMVX_DIM: Info TCID 0, avg_frame_jitter =
0 ->
```

## udpstatShow

Syntax:

**`udpstatShow`**

Displays the UDP protocol statistics.

```
-> udpstatShow
UDP:
2880748 total packets
1443277 input packets
1437471 output packets
0 incomplete header
0 bad data length field
0 bad checksum
4186 broadcasts received with no ports
```

```
0 full socket
1433985 pcb cache lookups failed
968 pcb hash lookups failed
value = 29 = 0x1d
```

### uftpFwDnldMon

Syntax:

**`uftpFwDnldMon "IPAddr"`**

Debug the UFTP IP Telephone Firmware Download if you encounter a problem. This command provides detailed information about the UFTP download operation and helps you locate the source of the problem.

The following table describes the command parameters.

**Table 70**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| IPAddr | "string" | Optional. IP address of the telephone to monitor. |

To enable the download monitor function, enter a valid IP address for the IP Phone. Example enabling the UFTP monitor for an IP Phone:

```
-> uftpFwDnldMon "47.11.217.20"
UFTP F/W Download monitor is enabled to monitor set with IP
47.11.217.20
```

To disable the monitor function, enter the command without passing any parameter.

The following example shows the output for disabling the UFTP monitor.

```
-> uftpFwDnldMon Input is empty.  UFTP F/W Download monitor
is disabled.
```

### uftpNodeShow

Syntax:

**`uftpNodeShow`**

Provide a complete UFTP IP Telephone Firmware Download Summary of each node. The output includes the cards in the node that are configured but do not respond.

Each node summary contains the following information:

- Index
- TN (in format l s c u)

- Host Type
- TLAN IP Address
- Active Download Count (Act)
- Server Up Time (Srv Up Time)
- Successful Download Count (Ok)
- Failure Download Count (Fail)

The following example shows command output.

```
oam> uftpNodeShow

Retrieving information form the peer(s), please wait!

--------- UFTP IP Phone Firmware Download Summary for Node
5488 ---------
Index TN Host Type TLAN IP Addr Act Srv Up Time Ok Fail
01 ISP 1100 47.  11.213.  83 002 0000 01:36:12 00070 00001
02 100 1 15 SMC 47.  11.213.  79 001 0000 02:25:10 00050
00001
03 20 1 2 ITG-P 47.  11.213.103 001 0000 05:23:10 00048 00001
------------------------------------------------------------
-------------------------------------------------
Total 004 00168 00003
------------------------------------------------------------
--------------------------------------------------
------------- card in node configured that are not
responding ------------------
Index TN Host Type TLAN IP Addr
04 20 1 7 SMC 47.  11.213.  158
------------------------------------------------------------
-------------------------------------------------
```

## uftpShow

Syntax:
**`uftpShow`**

This command provides the following information:

- configuration information about the UFTP
- count of successful downloads after they start since the Signaling Server/SMC reboot
- count of downloads that failed or prematurely ended after they start since the Signaling Server/SMC reboot
- number of active downloads and list for each

- Terminal Type
- IP Address of the telephone being download
- number of bytes sent for the download

The following example shows command output.

```
-> uftpShow
------ UFTP Server Configuration ----------------------
UFTP server IP address ........... 47.11.24.158 (port :
5105)
Concurrent downloading limit ....... 15(sets)

Total IP Set firmware = 5

FirmWare TermType PolicyName FileName
---------- ---------- ------------ ---------------
0602B59 i2004 DEFAULT_I2004 /ums/i2004.fw
0603B59 i2002 DEFAULT_I2002 /ums/i2002.fw
0604C00 i2001 DEFAULT_IPPH2 /ums/IPP2SETS.fw
0604C00 i2002 Ph2 DEFAULT_IPPH2 /ums/IPP2SETS.fw
0604C00 i2004 Ph2 DEFAULT_IPPH2 /ums/IPP2SETS.fw

-------------- Run Time Data -------------
Last UFTP reset ........................ 19/12/03
18:50:18
Cumulation Period ......................... 0000
19:07:22
Successful downloads .........................147
Fail downloads ......................... 20
------------- Active downloads ---------
Current downloading sets 5
TermType IP Address Downloaded[KByte]
--------- ---------- -----------------
i2004 47.11.2.157 122
i2004 47.11.2.168 71
i2004 47.11.2.215 41
i2002 47.11.5.157 26
i2001 47.11.3.158 15
```

## uftpRunTimeDataReset

Syntax:
**`uftpRunTimeDataReset`**

Reset the run time data field in the UFTP data block.

The following example shows command output.

```
oam> uftpRunTimeDataReset

Run time data reset OK.
----------------- Run Time Data ----------------
Successful downloads .......................... 400
Fail downloads ......................... 0
```

## UKLossPlanClr

Syntax:

**`UKLossPlanClr`**

Clear the gain adjustment made by the UKLossPlanSet and lossPlanSet commands and returns the IP Phones to the TIA-810A levels. This command is an alias to the lossPlanClr command.

Run this command on the node Leader card while it is the node master to ensure that the data propagates correctly to all cards in the node. When you install a new leader card on a node with modified levels, always enter the loss plan command on the CLI even if you previously entered the command on the CLI of another card.

The following example shows command output.

```
VGMC> UKLossPlanClr
value = 0 = 0x0
VGMC>
IP client loss plan set to default values
```

## UKLossPlanSet

Syntax:

**`UKLossPlanSet`**

Increase the handset and headset gain settings in the IP Phone by 5 dB to raise the volume levels beyond those specified by TIA-810A/TIA-912 to be closer to those used for the digital phones in the United Kingdom. No parameters are needed, as this command increases the gain by a fixed amount.

Run this command on the node Leader card while it is the node master, to ensure that the data propagates correctly to all cards in the node. When you install a new leader card on a node with modified levels, always enter the loss plan command on the CLI even if you previously entered the command on the CLI of another card.

After you enter this command, the gain adjustment is downloaded to all registered telephones. When phones register, they are downloaded the new gain values. The gain adjustment is also saved to a disk file (/c:/config/loss.ini) so the adjusted gains are retained when the card reboots.

When a node has a modified loss plan (that is, either this command or lossPlanSet was used), a new card added to the node is updated with the modified loss plan 30 seconds after it boots up. Prior to that being received, calls made by IP Phones registered to the new card have the default loss plan levels.

> ***Note:*** Systems with this command activated may experience increased occurrence of echo and other audio issues related to the increased volume.

The following example shows command output.

```
VGMC> UKLossPlanSet
value = 0 = 0x0
VGMC>
IP client loss plan adjusted to UK levels
```

### umsKernelShowJobs

Syntax:
**`umsKernelShowJobs "jobType"`**

Print current active and pending firmware update jobs. This is a useful consolidation of information; the prior version of the command displayed details of each IP Phone in the job list on the Signaling Server where up to 1000 IP Phones are registered.

The following table describes the command parameters.

**Table 71**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| jobType | active<br>pending<br>wait | Optional. If not included, all jobs are displayed. If a job type is specified, then only those jobs are displayed. |

The following example shows the output with all jobs displayed.

```
-> umsKernelShowJobs
=================================
Kernel Job Summary
=================================
Jobs:
Tag timeStamp terminalID Firmware ipAddress terminalType
jobStatus waitReason NewFirmware retryCount
---------- --------- ---------- -------- ---------------
------------ --------- --------- -------- ----------
0x022db908 1035383728 0x0230c13c 0602B38 47.11.243.198
i2004 Waiting Aging 0602B40 0
0x022db878 1035383727 0x02308d10 0602B38 47.11.243.194
i2004 Waiting Aging 0602B40 0 0x02bbd6c4 1035383729
0x022ee928 0602B39 47.11.243.190
i2004 Waiting Aging 0602B40 0 0x02bbd634 1035383724
0x022f0018 0602B39 47.11.243.183
i2004 Waiting Aging 0602B40 0 0x02bbd5a4 1035383729
0x02309254 0602B38 47.11.243.197
i2004 Waiting Aging 0602B40 0 0x02bbd484 1035383724
0x022fe800 0602B38 47.11.243.97
i2004 Waiting Aging 0602B40 0 0x02bbd3f4 1035383728
0x022fe9d0 0602B38 47.11.243.195
i2004 Waiting Aging 0602B40 0 0x02bbd2d4 1035383714
0x022f7590 0603B39 47.11.243.31
i2002 Active Noreason 0603B40 0 0x02bbd208 1035383715
0x022e5dd0 0602B39 47.11.243.196
i2004 Active Noreason 0602B40 0 0x02bbd13c 1035383716
0x0231042c 0602B38 47.11.243.185
i2004 Active Noreason 0602B40 0 0x02bbd0ac 1035383724
0x022e53f0 0602B39 47.11.243.186
i2004 Active Noreason 0602B40 0 0x02bbcf70 1035383724
0x02e21aa0 0602B38 47.11.243.33
i2004 Active Noreason 0602B40 0 0x02bbcee0 1035383727
0x0231148c 0602B38 47.11.243.224
i2004 Active Noreason 0602B40 0
--------------
Total Jobs=12

Active Main:
Server MaxClients JobTags
----------------- --------- -------
47.11.243.176 10 6
--------------
Total Server Queue Tiers=1

Wait Main:
Waitreason JobTags
---------- -------
Aging 6
--------------
Total Waiting Queue Tiers=1
```

```
Pending Main:
TerminalType Firmware JobTags
------------ -------- -------
i2002 0603B40 0
i2004 0602B40 0
--------------
Total Pending Queue Tiers=2
value = 45 = 0x2d = '-'
->
```

## umsPolicyShow

Syntax:

**`umsPolicyShow`**

Displays information pertaining to the download of IP Phone firmware. Although the example is shown for the VGMC, it is applicable from the Signaling Server vxshell prompt.

The following example shows command output.

```
VGMC> umsPolicyShow

Total firmware = 2

FirmWare TermType PolicyName Server FileName Limit When
Upgrade Protocol Retry
---------- -------- ------------- ---------------- ------
--------- ----- --------- ---------- --------------- -----
0602B38 i2004 DEFAULT_I2004 192.168.1.140 /ums/i2004.fw 10
ALWAYS ANY TFTP -1
3002B20 i2002 DEFAULT_I2002 192.168.1.140 /ums/i2002.fw 10
NEVER ANY TFTP -1
value = 0 = 0x0
```

## umsSetFirmwarePolicy

Syntax:

**`umsSetFirmwarePolicy "fwVersion" "policy" "when" "upgradeType"`**

Enable the parameters of a UMS firmware upgrade policy to change.

The following table describes the command parameters.

**Table 72**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| fwVersion | – | The firmware version to use in the upgrade. |
| policy | DEFAULT_I2004<br>DEFAULT_I2002<br>DEFAULT_IPPH2 | The ums upgrade policy to edit. |
| when | NEVER<br>ALWAYS<br>IDLE | When the upgrade is to perform. |
| upgradeType | UPGRADE<br>DOWNGRADE<br>ANY | The direction considered for the upgrade. |

The following example shows command output.

```
-> umsSetFirmwarePolicy "0302B59", "MINIMUM", "IDLE",
"UPGRADE"
value = 1 = 0x1
->
```

### umsUpdatePolicy

Syntax:
**`umsUpdatePolicy`**

When you manually copy a firmware file to a card, this command notifies the UMS task of the new file so it can be processed.

The following example shows command output.

```
-> umsUpdatePolicy
value = 0 = 0x0
->
OCT 16 16:00:04 tShell:  Info Resync IniFile from version
0602B38 to 0602B38 for terminal i2004.
```

### umsUpgradeAll

Syntax:
**`umsUpgradeAll`**

Initiate a firmware download to all IP Phones requiring a firmware upgrade. In the following example, both regisatered IP Phones already have the latest firmware and thus are not upgraded. Although the example is shown for the VGMC, it is applicable from the Signaling Server vxshell prompt.

> ***Note:*** You must manually install the PC software for the IP Softphone 2050. The VGMC/Signaling Server application does not check the firmware for the IP Softphone 2050.

The following example shows command output.

```
VGMC> umsUpgradeAll
value = 0 = 0x0
VGMC>
FEB 16 15:25:36 tShell:  Info ums Session download all
FEB 16 15:25:36 tUMS: Info try to upgrade all registered
sets
FEB 16 15:25:36 tSET: Info Terminal offline 192.168.1.141
TN 0x6005
FEB 16 15:25:36 tSET: Info Terminal offline 192.168.1.142
TN 0x6006
FEB 16 15:25:36 tUMS: Info decision-bless "192.168.1.141"
FEB 16 15:25:36 tUMS: Info decision-bless "192.168.1.142"
FEB 16 15:25:36 tSET: Info 192.168.1.141 TN 61-01
Registered with M1
FEB 16 15:25:36 tSET: Info 192.168.1.142 TN 61-02
Registered with M1
VGMC>
```

## umsUpgradeTimerCancel

Syntax:

**`umsUpgradeTimerCancel`**

Cancel the firmware upgrade timer if it is running.

The following example shows the output from VGMC or Signaling Server vxshell prompt.

```
VGMC> umsUpgradeTimerCancel
value = 0 = 0x0
VGMC> umsUpgradeTimerShow
Upgrade Timer is not active.
value = 30 = 0x1e
```

## umsUpgradeTimerSet

Syntax:

**`umsUpgradeTimerSet delay`**

Assign the delay, in seconds, from the current time to when the IP Phone firmware upgrade will occur.

The following table describes the command parameters.

**Table 73**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| delay | – | Seconds of delay before the firmware upgrade occurs. |

```
-> umsUpgradeTimerSet 3600
value = 0 = 0x0
->
FEB 16 14:26:49 tShell:  Info Upgrade will happen after 3600
seconds
->
```

## umsUpgradeTimerShow

Syntax:
**`umsUpgradeTimerShow`**

Displays the time set for a firmware upgrade. Although the example is shown for the VGMC, it is applicable from the Signaling Server vxshell prompt.

The following example shows command output.

```
VGMC> umsUpgradeTimerShow
Now UpgradeTime TimeDiff
-------------------- -------------------- ----------
FEB 16 02:27:47PM FEB 16 03:26:49PM 0:59:02
value = 53 = 0x35 = '5'
```

## upgradeErase

Syntax:
**`upgradeErase`**

Erase the application binaries from the ITG-P card flash memory. You can run this command from the BIOS or VxWorks prompt.

This command does not apply to the SMC or Signaling Servers.

The following example shows command output.

```
-> upgradeErase
value = 0 = 0x0
-> Erasing....
Bank 1...0xf9800000, sector 0.
Bank 2...0xf9900000, sector 8.
Bank 3...0xf9a00000, sector 16.
Bank 4...0xf9b00000, sector 24.
A restart will boot from the BIOS ROMs
```

## usiGainTableShow

Syntax:

**`usiGainTableShow transducer`**

Print the lookup table used to map the dB loss values calculated by the MAM task into the appropriate gain settings for the DSP and CODEC on the IP Phone. Each output line comprises four values: index (provided for reference), loss (dB), CODECgain, and DSPgain.

The following table describes the command parameters.

**Table 74**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| transducer | 1–3 | Specify the transducer gain table to print.<br>1 =Handset<br>2 =Headset<br>3 = Handsfree |

The following example shows the output for the handset lookup table.

```
-> usiGainTableShow 1

=== RLR Map ===
30 12.0 0x31 0x00 31 11.5 0x30 0x00 32 11.0 0x2f 0x00 33 10.5
0x2e 0x00 34 10.0 0x2d 0x00
35 9.5 0x2c 0x00 36 9.0 0x2b 0x00 37 8.5 0x2a 0x00 38 8.0 0x29
0x00 39 7.5 0x28 0x00
40 7.0 0x29 0x00 41 6.5 0x28 0x01 42 6.0 0x29 0x02 43 5.5 0x28
0x02 44 5.0 0x29 0x03
45 4.5 0x28 0x03 46 4.0 0x29 0x04 47 3.5 0x28 0x04 48 3.0 0x29
0x05 49 2.5 0x28 0x05
50 2.0 0x29 0x06 51 1.5 0x28 0x06 52 1.0 0x29 0x07 53 0.5 0x28
0x07 54 0.0 0x29 0x08
55 -0.5 0x28 0x08 56 -1.0 0x29 0x09 57 -1.5 0x28 0x09 58 -2.0
0x29 0x0a 59 -2.5 0x28 0x0a
60 -3.0 0x29 0x0b 61 -3.5 0x28 0x0b 62 -4.0 0x29 0x0c 63 -4.5
0x28 0x0c 64 -5.0 0x29 0x0d
```

```
65 -5.5 0x28 0x0d 66 -6.0 0x29 0x0e 67 -6.5 0x28 0x0e 68 -7.0
0x29 0x0f 69 -7.5 0x28 0x0f
70 -8.0 0x29 0x10 71 -8.5 0x28 0x10 72 -9.0 0x29 0x11 73 -9.5
0x28 0x11 74 -10.0 0x29 0x12

=== SLR Map ===
70 17.5 0x00 0x02 71 17.0 0x01 0x03 72 16.5 0x00 0x00 73 16.0
0x01 0x04 74 15.5 0x00 0x04
75 15.0 0x01 0x05 76 14.5 0x00 0x05 77 14.0 0x01 0x06 78 13.5
0x00 0x06 79 13.0 0x01 0x07
80 12.5 0x00 0x07 81 12.0 0x01 0x08 82 11.5 0x00 0x08 83 11.0
0x01 0x09 84 10.5 0x00 0x09
85 10.0 0x01 0x0a 86 9.5 0x00 0x0a 87 9.0 0x01 0x0b 88 8.5
0x00 0x0b 89 8.0 0x01 0x0c
90 7.5 0x00 0x0c 91 7.0 0x01 0x0d 92 6.5 0x00 0x0d 93 6.0 0x01
0x0e 94 5.5 0x00 0x0e
95 5.0 0x01 0x0f 96 4.5 0x00 0x0f 97 4.0 0x01 0x10 98 3.5 0x00
0x10 99 3.0 0x01 0x11
100 2.5 0x00 0x11 101 2.0 0x01 0x12 102 1.5 0x00 0x12 103 1.0
0x01 0x13 104 0.5 0x00 0x13
105 0.0 0x01 0x14

=== STMR Map ===
00 12.0 0x00 0x00 01 15.0 0x00 0x00 02 18.0 0x00 0x00 03 21.0
0x00 0x00 04 24.0 0x00 0x00
05 27.0 0x00 0x00 06 30.0 0x00 0x00 07 33.0 0x00 0x00
```

## usiLibTraceHelp

Syntax:

**`usiLibTraceHelp`**

This command displays syntax and information for available commands.

## usiLibTraceSettings

Syntax:

**`usiLibTraceSettings`**

Print the current trace settings.

## usiTraceSetOutput

Syntax:

**`usiTraceSetOutput trace_output, "file_name"`**

Assign the output destination for the trace tool.

The following table describes the command parameters.

**Table 75**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| trace_output | 1–4 | Trace output destination, where<br>1 = TTY<br>2 = SYSLOG<br>3 = File<br>4 = File and TTY |
| file_pathname | "string" | File to output to if trace_output = 3 or 4.<br><br>You can enter the entire path name or only the file name. If you enter only the file name, the default path is C:/file_name. Enclose the string in quotation marks. |

The following example shows command output.

```
-> usiTraceSetOutput 1, "/C:/trace.txt"
value = 0 = 0x0
```

## usiLibTraceOff

Syntax:
**`usiLibTraceOff "ipAddr"`**

This command turns off the trace for the specified IP Address of a telephone.

The following example shows command output.

```
-> usiLibTraceOff "47.11.213.216"
value = 0 = 0x0
```

## usiLibTraceAllOff

Syntax:
**`usiLibTraceAllOff`**

This command turns off the trace for all IP addresses (for all telephones).

## usiLibTraceOn

Syntax:
**`usiLibTraceOn "ipAddr" , to_set, from_set`**

Turn on the trace for one telephone.

The following table describes the command parameters.

**Table 76**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| to_set | – | Message types sent to the telephone to trace, where<br>0 = Off<br>1 = Broadcast Manager messages<br>2 = Audio Manager messages<br>4 = Display Manager messages<br>8 = Key/Indicator Manager messages<br>16 = Basic Manager messages<br>32 = Network Manager messages<br>64 = Accessory Manager messages<br>128 = Accessory Devices messages<br>255 = All messages<br><br>To track multiple message types but not all message types, add together the values for the message types you wish to track (for example, to track Key/Indicator Manager and Display Manager messages, add 4+8, and assign to_set a value of 12). |
| from_set | – | Message types to trace from the telephone, where<br>0 = Off<br>1 = Broadcast Manager messages<br>2 = Audio Manager messages<br>4 = Display Manager messages<br>8 = Key/Indicator Manager messages<br>16 = Basic Manager messages<br>32 = Network Manager messages<br>64 = Accessory Manager messages<br>128 = Accessory Devices messages<br>255 - All messages<br><br>To track multiple message types but not all message types, add together the values for the message types you wish to track (for example, to track Key/Indicator Manager and Display Manager messages, add 4+8, and assign to_set a value of 12). |

The following example shows the output for tracing network and basic manager messages sent to the telephone displayed on the TTY.`->`

```
usiLibTraceOn "47.11.213.216", 48, 0
value = 0 = 0x0
```

## usiQueryAPB

Syntax:

**`usiQueryAPB SetBlkAddr, transducer`**

This command triggers a query to the specified IP Phone and retrieves the current levels for the specified transducer.

The various gain blocks on the IP Phone have the following step sizes:

- Receive (RX) CODEC gain (0.5 dB step)
- Tx CODEC gain (0.5 dB step)
- Receive (RX) DSP gain (1 dB step)
- Tx DSP gain (1 dB step)
- Sidetone gain (3 dB step)

You can look up the value for each gain block in the printed output to determine the dB level on the telephone for the transducer.

The following table describes the command parameters.

**Table 77**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| SetBlkAddr | – | Pointer value of the telephone given by the usiShow command. |
| transducer | 1–3 | Specify the transducer data to retrieve, where<br>1 = Handset<br>2 = Headset<br>3 = Handsfree |

The following example shows command output.

```
-> usiQueryAPB 0x03a876fc,1

value = 0 = 0x0
-> APB Number = 1
Return to Default Disabled
Automatic Gain Control Disabled
APB for Volume Control Enabled = 0x03
Listener Sidetone Enabled
Acoustic Echo Canceller Disabled
Step Size = 0x00
Max Volume = 0x08
Min Volume = 0x0b
Rx Codec Gain = 0x29
Tx Codec Gain = 0x01
Rx DSP Gain = 0x06
Tx DSP Gain = 0x09
Sidetone Gain = 0x02
```

```
AEC Length in number of taps = 0x00 0x80
Microphone noise threshold = 0x19
Line Delay Length = 0x01
Max Return Loss TG = 0x0c
Switched Loss when AEC Off = 0x08
NormDelta = 0x0c
TxREF_LEVEL Compensation when AEC Off = 0x04
TxREF_LEVEL Compensation when AEC On = 0x08
Noise Wait Counter = 0x4b
APS (Audio Processing Shell) = 0x48 0x0b 0x50 0x06 0x32 0x80
0xc4 0x01
0x11 0x0c 0x57 0x08 0x40 0x00 0x24 0x07 0x02 0x89 0x28 0x00
0x00 0x4b
Default Volume = 0x01
Current Volume = 0x01
Sampling Rate = 0x00
HIP Filter = 0x00
AGC Threshold = 0x17
LST Threshold = 0x0e

-> usiQueryAPB 0x03a876fc,2
value = 0 = 0x0

-> APB Number = 2
Return to Default Disabled
Automatic Gain Control Disabled
APB for Volume Control Enabled = 0x03
Listener Sidetone Enabled
Acoustic Echo Canceller Disabled
Step Size = 0x00
Max Volume = 0x08
Min Volume = 0x0b
Rx Codec Gain = 0x25
Tx Codec Gain = 0x04
Rx DSP Gain = 0x10
Tx DSP Gain = 0x01
Sidetone Gain = 0x01
AEC Length in number of taps = 0x00 0x80
Microphone noise threshold = 0x19
Line Delay Length = 0x01
Max Return Loss TG = 0x0c
Switched Loss when AEC Off = 0x0c
NormDelta = 0x04
TxREF_LEVEL Compensation when AEC Off = 0x04
TxREF_LEVEL Compensation when AEC On = 0x08
Noise Wait Counter = 0x4b
APS (Audio Processing Shell) = 0x0b 0x8b 0x50 0x10 0x32 0x80
```

```
0xc4 0x01 0x11
0x84 0x57 0x08 0x40 0x00 0x24 0x07 0x02 0x89 0x28 0x00 0x00
0x4b
Default Volume = 0x01
Current Volume = 0x01
Sampling Rate = 0x00
HIP Filter = 0x00
AGC Threshold = 0x17
LST Threshold = 0x0e

-> usiQueryAPB 0x03a876fc,3
value = 0 = 0x0

-> APB Number = 3
Return to Default Disabled
Automatic Gain Control Disabled
APB for Volume Control Enabled = 0x03
Listener Sidetone Disabled
Acoustic Echo Canceller Disabled
Step Size = 0x00
Max Volume = 0x08
Min Volume = 0x0b
Rx Codec Gain = 0x49
Tx Codec Gain = 0x08
Rx DSP Gain = 0x1a
Tx DSP Gain = 0x07
Sidetone Gain = 0x00
AEC Length in number of taps = 0x01 0x80
Microphone noise threshold = 0x14
Line Delay Length = 0x02
Max Return Loss TG = 0x0d
Switched Loss when AEC Off = 0x1a
NormDelta = 0x08
TxREF_LEVEL Compensation when AEC Off = 0x0c
TxREF_LEVEL Compensation when AEC On = 0x01
Noise Wait Counter = 0x4b
APS (Audio Processing Shell) = 0x3b 0x8b 0xdf 0x5a 0x29 0x80
0xd2 0x42 0x33
0x48 0x57 0xc1 0x5d 0x91 0x04 0xc5 0x02 0x89 0x28 0xff 0x00
0x4b
Default Volume = 0x01
Current Volume = 0x01
Sampling Rate = 0x00
HIP Filter = 0x00
AGC Threshold = 0x17
LST Threshold = 0x00
->
```

## usiQueryResetReason

Syntax:

**`usiQueryResetReason "phone IP address"`**

Send a request to the IP Phone as to the reason for the last reset. The response from the IP Phone appears on the console. This command is supported only by IP Phone firmware Release 1.26 or later.

The following table describes the command parameters.

**Table 78**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| phone IP address | "string" | IP address of the IP Phone. Enclose the string in quotation marks. |

The following example shows the output where the query is not supported by the IP Phone (firmware 1.25).

```
-> isetShow
IP Address Type State Up Time TN HWID FWVsn SrcPort
------------------ ------- ----------- ------------
------------ ------------------ ------- -------
192.168.1.143 i2004 online 1 01:29:21 061-03 1800802ddcd4c
66600 3002B25 5100

Total sets = 1
value = 0 = 0x0
->
-> usiQueryResetReason "192.168.1.143"
value = 0 = 0x0
->
FEB 28 15:15:57 VTM: Info 192.168.1.143:  Last Reset Reason
127 (Not Supported)
```

The following example shows the output where the first IP Phone queried has power cycled to it (that is, the adapter is unplugged), and the second IP Phone had been trying to register for some time prior to being connected to the TPS.

```
-> isetShow
IP Address Type State Up Time TN HWID FWVsn SrcPort
------------------ ------- ----------- ------------
------------ ------------------ ------- -------
192.168.1.142 i2004 online 0 00:12:33 061-02 180060387621a
96600 C902B26 5100
```

```
192.168.1.141 i2004 online 1 01:12:32 061-01 180060387621a
f6600 C902B26 5100
Total sets = 2
value = 0 = 0x0
->
-> usiQueryResetReason "192.168.1.142"
value = 0 = 0x0
->
FEB 28 15:18:17 VTM: Info 192.168.1.142:  Last Reset Reason
0 (Power up)

-> usiQueryResetReason "192.168.1.141"
value = 0 = 0x0
->
FEB 28 15:19:01 VTM: Info 192.168.1.141:  Last Reset Reason
2 (Soft reset:  RUDP retry attempts have been exhausted)
```

The following table describes possible values for the Reset Reason parameter for this command.

**Table 79**
**Reset Reason values**

| Code | Reset reason | SYSLOG output message | Notes |
|---|---|---|---|
| 0xff | Not Supported | Last Reset Reason Not Supported | – |
| 0x7f | Not Supported | Last Reset Reason Not Supported | – |
| 0 | Power up: code =0, Also covers $ | Power up | – |
| 1 | Soft reset: Watchdog timeout | Soft Reset: Watchdog timeout | – |
| 2 | Soft reset: RUDP retry attempts have been exhausted | Soft Reset: RUDP retry exhausted | – |
| * | Soft reset: by UNISTIM from TPS (network.c:116) | – | Does not need to be recorded, or cannot be recorded. |
| * | Soft reset: by server switch (network.c:280) | – | Does not need to be recorded, or cannot be recorded. |

**Table 79**
**Reset Reason values (cont'd.)**

| Code | Reset reason | SYSLOG output message | Notes |
|---|---|---|---|
| 5 | Hard reset: by UNISTIM from TPS (Ged.c:995) or by a DSP interrupt reset caused by SKS used to reset the phone ([mute][up][down][up][down][up][9][release]) or bad AB06 chip) (Ged.c:995) | Hard reset by TPS | – |
| $ | Hard reset: by DTEV UART DSR interrupt (DTEVuart.c:585) | DTEV UART DSR interrupt | Cannot be recorded, and is covered by Reset Reason 5. |
| 7 | Hard reset: by software upgrade: command had bad data (network.c:571) | Upgrade: command had bad data | – |
| # | Hard reset: by software upgrade: unknown download protocol (not TFTP or UFTP) (dwnloadr.c:124) | Upgrade: unknown download protocol | Can never happen or is covered by other error codes. |
| 9 | Hard reset: by software upgrade: read from TFTP/UFTP failed. For example, server is stopped in middle of downloading (dwnloadr.c: 207) | Upgrade: TFTP/UFTP failed. | – |
| 10 | Hard reset: by software upgrade: unrecognized Flash Manufacturer ID (dwnloadr.c:251) | Upgrade: Bad Flash Manufacturer Id | – |
| 11 | Hard reset: by S/W upgrade: download finished | Upgrade: download finished | – |
| 12 | Hard reset: by DHCP: IP Phone has bound an IP address, but the MAC address in ACK message does not match the IP Phone (cl_dhcp.c:915) | DHCP: Bad MAC in ACK Msg | – |
| 13 | HardReset: by DCHP: Bad ACK parameter, like FULL DHCP, but parameter is bad (cl_dhcp.c:968) | DHCP: Bad parameter in ACK Msg | – |

**Table 79**
**Reset Reason values (cont'd.)**

| Code | Reset reason | SYSLOG output message | Notes |
|---|---|---|---|
| 14 | HardReset: by DHCP: Address lost, lease is up or extension refused. (cl_dhcp.c:996) | DHCP: Address lost | – |
| 15 | Hard Reset: by DHCP: Address failed (cl_dhcp.c:1033) | DHCP: Add address failed | – |
| 16 | Hard Reset: IP check: duplicated IP address (Winmgr.c:925) | Duplicated IP address | – |
| 17 | Hard Reset: IP check: cannot check IP address (winmgr.c:929) | cannot check IP address | – |
| 18 | Hard Reset: Startup: Current Server is secondary, retry attempts have been exhausted | Startup: S2 Retry Exhausted | – |
| # | Hard reset: Startup: Primary server action code is invalid (winmgr.c:1231) | Startup: Invalid S1 | – |
| # | Hard reset: Startup: Secondary server action code is invalid (winmgr.c:1238) | Startup: S2 action code is invalid | – |
| 21 | Hard reset: Startup: Manual Configuration: Softkey 3: CANCEL function (winmgr.c:2099) | Startup: Manual config cancelled | – |
| 22 | Hard reset: Close Audio Stream, pSOS cannot delete the tx task | O/S: Cannot delete Tx Task | – |
| 23 | Hard reset: Close Audio Stream, pSOS cannot delete the rx task. | O/S: Cannot delete Rx task | – |

### usiShow

Syntax:

```
usiShow
```

Print terminal-related information. You can use this command to obtain the IP Phone block address (SetBlkAddr) for a telephone. The parameter SetBlkAddr is a pointer to a data structure for the telephone used by the command usiQueryAPB.

The output data is broken into three blocks: general information, a list of ports, and a list of terminals with active RUDP connections.

The following table describes the data parameters printed by this command.

**Table 80**
**Data output**

| Parameter | Description |
|---|---|
| General information parameters | |
| Num of sets | Number of telephones with RUDP links on this card. |
| DebugTrace | On/Off indication of UNIStim message trace (see the usi commands). |
| Tos byte | Value of ToS byte sent in RUDP messages |
| RudpWinSize | Number of outstanding RUDP frames allowed before an Ack is received. Configurable in the tps.ini file. For more information about configuring this parameter, see "RUDP windowing control" (page 418). |
| UsiSetList | Pointer to head of terminal list. |
| UsiPortList | Pointer to head of port list. |
| Port parameters | |
| Port ID | Software ID of a port. Is the pointer to the port control block. |
| IPAddress | IPAddress used by the port. |
| Port | UDP port number. |
| TaskID | Internal number identifying the task that processes the messages from the port |
| Terminals | Number of terminals messaging with the port. |
| Terminal parameters | |
| Set | Software ID of a port. Is the pointer to the port control block. |
| SetBlkAddr | Pointer to the telephone control block. |
| Connect | Yes/No whether the IP Phone RUDP link is active. |

**Table 80**
**Data output (cont'd.)**

| Parameter | Description |
|---|---|
| ConnectID | ID of the IP Phone RUDP connection. |
| FwVsn | Firmware version on the terminal.Firmware version on the terminal. |
| IPAddress | IP Address of the terminal. |
| SrcPort | UDP port used on the TPS card for the RUDP link. |
| DestPort | UDP port used on the terminal for the RUDP link. |

You can modify the RudpWinSize parameter by changing the TPS.INI file. For more information, see "RUDP windowing control" (page 418).

The following example shows command output.

```
-> usiShow
usiLib
======
Num of sets 2
DebugTrace Off
Tos byte 0x00
RudpWinSize 10
UsiSetList 0x003b6064
UsiPortList 0x003b6070

PortID IPAddress Port TaskID Terminals
---------- -------------- ---- ------ ---------
0x037f7abc 192.168.1.140 5100 0x7800 2
0x037e2cdc 192.168.1.140 7300 0x9800 0
0x03a88398 192.168.1.149 4100 0xa800 0

Set SetBlkAddr Connect ConnectID FwVsn IPAddress SrcPort
DestPort
--- ---------- ------- ---------- ------- ---------------
------- --------
1 0x03a87504 yes 0x03a87584 3002B00 192.168.1.141 5100 5000
2 0x03a87760 yes 0x03a877e0 3002B00 192.168.1.142 5100 5000
value = 0 = 0x0
->
```

### vgwAudioTraceHelp

Syntax:

**`vgwAudioTraceHelp`**

This command describes the CLI commands and required parameters.

### vgwAudioTraceSettings

Syntax:

**`vgwAudioTraceSettings`**

Print the current trace settings.

The following example shows command output.

```
-> vgwAudioTraceSettings
Monitor is off
value = 0 = 0x0
```

### vgwAudioTraceAllOff

Syntax:

**`vgwAudioTraceAllOff`**

Turn off the trace for all channels.

### vgwAudioTraceoff

Syntax:

**`vgwAudioTraceoff chNum`**

Turn off the trace for the specified channel.

The following table describes the command parameters.

**Table 81**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number |

### vgwAudioTraceOn

Syntax:

**`vgwAudioTraceOn chNum`**

Initiate audio message tracing. The channel number specifies which gateway or channel on which to perform ACP or audio message tracing.

The following table describes the command parameters.

**Table 82**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number |

The following example shows command output.

```
-> vgwAudioTraceOn 10
value = 0 = 0x0
```

### vgwAudioTraceSetOutput

Syntax:

**`vgwAudioTraceSetOutput trace_output, "file_pathname"`**

Assign the output destination for the trace tool.

The following table describes the command parameters.

**Table 83**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| trace_output | 1–4 | Trace output destination, where:<br>1 = TTY<br>2 = SYSLOG<br>3 = File<br>4 = File and TTY |
| file_pathname | "string" | Specify the file to output to if trace_output = 3 or 4. Enclose the string in quotation marks.<br><br>You can enter the entire path name, or the file name only. If you enter only the file name, the default path is "C:/file_name". |

The following example shows command output.

```
-> vgwAudioTraceSetOutput 3, "trace.txt"
value = 0 = 0x0
```

### vgwChStat

Syntax:

**`vgwChStat chNum`**

Print the statistics for a channel on the SMC card. This command applies only to the SMC card. The MAC layer statistics, both the general and channel-specific microEngine statistics are printed. If you assign a DimDspStat = 1 prior to using this command (see "DimDspStat = 1/0" (page 104)), the channel DSP statistics also print.

> ***Note:*** When DimDspStat is assigned a value of 1, the DSP statistics print at the end of every gateway channel call. Consider how busy the card is before you enable this, as a block of data prints every time a call completes until the variable is reset to 0. Be sure to assign DimDspStat = 0 after you use the vgwChStat command.

The following example shows command output.

```
-> vgwChStat 6
###Net Stats:  Base = 0x38410800#########
TxUniOkCnt = 7163 - The number of unicast packets
transmitted without any errors.
TxMltOkCnt = 0 - The number of multicast packets transmitted
without any errors.
TxBrdOkCnt = 54 - The number of broadcast packets
transmitted without any errors.
TxDeferCnt = 7163 - The number of packets deferred upon the
first transmit attempt due to a busy line.
TxColCnt = 0 - The number of regular collision events
occurring during transmission.
TxScolCnt = 0 - The number of packets transmitted without
any error following a single collision.
TxMcolCnt = 0 - The number of packets transmitted without
any error following multiple collisions.
TxXcolCnt = 0 - The number of packets that have experienced
16 consecutive collisions or more.
TxLcolCnt = 0 - The number of transmission abortions due to a
collision occurring after transmission of packets that are
64 bytes in length.
TxPkt64Cnt = 4818 - The number of transmitted packets, 64
bytes in length, including bad packets.
TxPkt65Cnt = 2351 - The number of transmitted packets, 65 to
127 bytes in length, including bad packets.
TxPkt128Cnt = 44 - The number of transmitted packets, 128 to
255 bytes in length, including bad packets.
TxPkt256Cnt = 0 - The number of transmitted packets, 256 to
511 bytes in length, including bad packets.
TxPkt512Cnt = 4 - The number of transmitted packets, 512 to
1023 bytes in length, including bad packets.
TxPkt1024Cnt = 0 - The number of transmitted packets, 1024
to 1518 bytes in length, including bad packets.
```

```
TxPkt1519Cnt = 0 - The number of transmitted packets with
length larger than 1519 bytes, including bad packets.
TxPauseCnt = 0 - The number of correct transmitted
flow-control packets.
TxErrCnt = 0 - The number of packets transmitted with
an error due to transmit FIFO underflow or txerr signal
assertion.
TxOctOkCnt(byte) = 513620 - The number of bytes transmitted
in good packets.
TxOctBadCnt(byte)= 0 - The number of bytes transmitted in
packets with errors.
RxOctOkCnt(byte) = 493504 - The number of bytes received in
good packets.
RxOctBadCnt(byte)= 0 - The number of bytes received in
packets with errors.
RxRuntCnt = 0 - The number of frames received without SFD
detection but with carrier assertion.  This counter must be
reset after moving to the SYM mode.
RxOvfCnt = 0 - The number of receive packets not fully
accepted due to receive FIFO overflow.
RxShortOkCnt = 0 - The number of packets, less than 64 bytes
in length, received without any error.
RxShortCrcCnt = 0 - The number of packets less than 64 bytes
in length, received with CRC error.
RxUniOkCnt = 7298 - The number of unicast packets with
lengths between 64 bytes and the maximum packet size,
received without any errors.
RxMltOkCnt = 0 - The number of multicast packets with
lengths between 64 bytes and the maximum packet size,
received without any errors.
RxBrdOkCnt = 13 - The number of broadcast packets with
lengths between 64 bytes and the maximum packet size,
received without any errors.
RxNormCrcCnt = 0 - The number of packets with lengths
between 64 bytes and the maximum packet size, received with
an integral number of bytes and a CRC error.
RxNormAliCnt = 0 - The number of packets with lengths
between 64 bytes and the maximum packet size, received with
a nonintegral number of bytes and a CRC error.
RxLongOkCnt = 0 - The number of packets, larger than the
maximum packet size, received without any error.
RxLongCrcCnt = 0 - The number of packets, larger than the
maximum packet size, received with a CRC error.
RxPkt64Cnt = 6248 - The number of received packets, 64 bytes
in length, including bad packets.
RxPkt65Cnt = 1064 - The number of received packets, 65 to 127
bytes in length, including bad packets.
```

```
RxPkt128Cnt = 0 - The number of received packets, 128 to 255
bytes in length, including bad packets.
RxPkt256Cnt = 0 - The number of received packets, 256 to 511
bytes in length, including bad packets.
RxPkt512Cnt = 0 - The number of received packets, 512 to 1023
bytes in length, including bad packets.
RxPkt1024Cnt = 0 - The number of received packets, 1024 to
1518 bytes in length, including bad packets.
RxPkt1519Cnt = 0 - The number of received packets, with
lengths between 1519 bytes and the maximum packet size
(programmable value), including bad packets.
RxPauseCnt = 0 - The number of correct received flow-control
packets.
RxFalsCrsCnt = 1 - The number of false carrier events
detected.
RxSymErrCnt = 0 - The number of received packets during
which PHY symbol errors were detected.
uengine dsp_rx_stat (packets to core) = 552 - The number of
packets that were sent to core.
uengine dsp_rx_stat (MAC21440 NO error) = 7318 - The number
of packets without errors.
uengine dsp_rx_stat (MAC21440 error) = 0 - The number of
packets with errors.
uengine dsp_rx_stat (broadcast packet) = 13 - The number of
broadcast packets.
uengine dsp_rx_stat (MAC address fail) = 0 - The number of
packets destined to other MACs.
uengine dsp_rx_stat (not IP) = 0 - The number of non IP
packets.
uengine dsp_rx_stat (not UDP) = 17 - The number of non UDP
packets.
uengine dsp_rx_stat (UDP port # out range) = 415 - The number
of UDP packets out of VGW port range.
uengine dsp_rx_stat (RTCP) = 120 - The number of RTCP
packets.  ( channel specific )
uengine dsp_rx_stat (SOP error) = 0 - The number of packets
with SOP errors.  (channel specific)
uengine dsp_rx_stat (source IP failure) = 0 - The number of
packets received from unknown senders.  (channel specific)
uengine dsp_rx_stat (active bit not set) = 0 - The number of
packets received while channel was not open.
uengine dsp_rx_stat (passed UDP port check) = 6775 - The
number of packets in VGW port range.
uengine dsp_rx_stat (packets to DSP) = 6776 - The number of
packets sent to the DSP. (channel specific)
uengine dsp_tx_stat (packets from DSP) = 6775 - The number
of packets received from DSP. (channel specific)
```

```
uengine dsp_tx_stat (invalid hdr, active bit not set) = 0
- The number of packets sent when the channel was not open.
(channel specific)
uengine dsp_tx_stat (packets to MAC) = 8234 - The number of
packets sent from micro engines to MAC.
MAR 25 03:47:34 tMVX_XSPY: Info 0000120288 - DIM: 1:2,
Tx='GET_RXTX_STAT'(67) clear=0x0
MAR 25 03:47:34 tMVX_XSPY: Info 0000120288 - DIM: 1:2, Rx
'GET_RXTX_STAT'(67) Len=57
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, rx_packet_count = 687
- number of received voice packets that are sent for playout
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, tx_packet_count =
2263 - number of transmitted voice packets
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, silence_packet_count
= 58553 - number of voice frames that were classified as
silence on a transmit side

MAR 25 03:47:34 tMVX_DIM: Info TCID 6, min_jitter = 10 -
minimum packet interarrival time
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, max_jitter = 33 -
maximum packet interarrival time
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, rtp_average_jitter =
0 - RTP average packet interarrival time
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, tx_grant_sync_drop =
0 - Number of frames dropped to align with a packet sync
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, tx_octets = 76288 -
Number of Tx octets
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, rx_octets = 36034 -
Number of Rx octets
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, aal2_cod_prof_chgs =
0 - Number of AAL2 codec profile def.
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, dtmf_tx_packets = 0 -
Number of Tx inband DTMF Relay packets
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, dtmf_rx_packets = 0 -
Number of Rx inband DTMF Relay packets
MAR 25 03:47:34 tMVX_XSPY: Info 0000120289 - DIM: 1:2,
Tx='GET_ERROR_STAT'(66) clear=0x0
MAR 25 03:47:34 tMVX_XSPY: Info 0000120289 - DIM: 1:2, Rx
'GET_ERROR_STAT'(66) Len=25
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, invalid_header_count
= 0 - number of incoming voice packets dropped due to invalid
header
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, to_micro_overflow_
count = 0 - number of transmit voice packets dropped due to
voice buffer overflow
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, rx_routing_dropped
= 0 - not used MAR 25 03:47:34 tMVX_DIM: Info TCID 6,
lost_enh_packet_count = 0 - number of lost incoming
enhancement voice packets
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, no_core_packet_count
= 0 - number of dropped incoming enhancement voice packets
due to absence of a core packet
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, pkt_lost_by_network
= 0 - packets lost on the network
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, rc4key_update_lost_p
kt_count = 0 - number of dropped packets due to RC4 key state
update
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, invalid_mac_heade
r_count = 0 - number of dropped packets due to invalid MAC
header
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, invalid_ssrc_count =
0 - number of dropped packets with invalid RX SSRC
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, invalid_payload_co
unt = 0 - number of dropped packets with invalid RX payload
type
MAR 25 03:47:34 tMVX_XSPY: Info 0000120290 - DIM: 1:2,
Tx='GET_VP_STAT'(65)
MAR 25 03:47:34 tMVX_XSPY: Info 0000120290 - DIM: 1:2, Rx
'GET_VP_STAT'(65) Len=21
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, avg_playout_delay =
24 - Average delay of VPU FIFO (in ms)
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, lost_packet_count = 0
- Lost Segment Counter
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, replay_packet_count
= 0 - Replay Segment Counter
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, idle_packet_count =
59989 - Idle Segment Counter
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, dropped_packet_count
= 9 - Drop Segment Counter
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, rx_packet_count =
8477 - Rx Segment Counter
MAR 25 03:47:34 tMVX_DIM: Info TCID 6, avg_frame_jitter = 1
- Average Packet Jitter (in ms) ->
```

## vgwPLLog

Syntax:

**`vgwPLLog state`**

This command controls the generation of log messages for packet loss detected during a call in the received packet stream (from the IP Phone). This command applies only to the VGMCs.

The following table describes the command parameters.

**Table 84**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| state | 0–2 | 0 = off<br>1 = print log message at call termination (default)<br>2 = print log messages as packet loss is detected during call<br><br>When state = 1 (the default setting), a message prints at the end of the call that shows the number of packets lost for the call duration and the percentage of the total packets sent. The call affected is identified by both the gateway channel and the terminal IP address.<br><br>When state = 2, a message prints each time packet loss is detected indicating the number of packets were lost at that moment. The call is identified by channel number. Changes to the state parameter are lost on power cycle or card reboot. |

The following example shows the output when state is equal to 1.

```
VGMC> vgwPLLog 1
value = 0 = 0x0
VGMC>
SEP 22 11:05:47 tRTP: Info channel 0 disconnected from
47.147.75.81, lost 6, total 430, precentageLost 1.4%, (tick
97514)
```

The following example shows the output when state is equal to 2.

```
VGMC> vgwPLLog 2
value = 0 = 0x0
VGMC>
SEP 22 11:05:49 tRTP: Info SML GAP in RTP seqNo chan 0 (recd:
22802, expect:  22800, sending 1001, gap 2, tick 98400)
```

## vgwRegisterTraceAllOff

Syntax:

**`vgwRegisterTraceAllOff`**

Turn off the trace for all channels.

## vgwRegisterTraceHelp

Syntax:

**`vgwRegisterTraceHelp`**

This command describes the CLI commands and parameters.

### vgwRegistrationTraceSetOutput

Syntax:

**`vgwRegistrationTraceSetOutput trace_output, "file_name"`**

The following table describes the command parameters.

**Table 85**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| trace_output | 1–4 | Trace output destination, where<br>1 = TTY<br>2 = SYSLOG<br>3 = File<br>4 = File and TTY |
| file_pathname | "string" | Specify the file to print to if trace_output = 3 or 4. Enclose the string in quotation marks.<br><br>You can enter the entire path name, or the file name only. If you enter only the file name, the default path is C:/file_name. |

The following example shows command output.

```
vgwTraceSetOutput 3, "trace.txt"
value = 0 = 0x0
```

### vgwRegistrationTraceOff

Syntax:

**`vgwRegistrationTraceOff chNum`**

Turn off the trace for the specified channel.

### vgwRegistrationTraceSettings

Syntax:

**`vgwRegistrationTraceSettings`**

Print the current trace settings.

### vgwShow

Syntax:

**`vgwShow "endpointIPAddr", port`**

Display status information for the VGMC active gateway channels. This command is not available on the Signaling Server.

If found, the display shows the identification of the card whose gateway channel connects to that address and shows the information for the channel. This is useful, for instance, when you need to identify from which card to collect gateway statistics (for example, packet loss). You can determine the same information from the Call Server using the TRAC/TRAK command in LD 80.

> ***Note:*** IP Phone to IP Phone calls do not appear because they do not use gateway channels. However, if an IP Phone is in a conference call (even with other IP Phones), a gateway channel is used for each telephone (because a TDM conference bridge is used).

The following table describes the command parameters.

**Table 86**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| endpointIPAddr | "string" | Optional. Specify the endpoint IP address to search for in all VGMC channels in the node. |
| port | – | Optional. Specify a port number to limit the output if multiple IP Phones share a single public signaling IP address (for example, behind a NAT device). |

The following table describes the data parameters printed by this command. Information prints only for active (nonidle and equipped) gateway channels.

**Table 87**
**Data output**

| Parameter | Description |
|---|---|
| Chan | Gateway channel |
| ChanState | State of channel (Idle, Busy, Disabled, or Unequipped) |
| DspMode | Mode DSP is in (Voice, Closed, Pending) |
| Codec | Frame size used for call |
| Tn | Physical TN of the channel in packed format |
| Reg | Status of channel gateway registration |
| Air Time | Duration of audio stream connection in seconds. A call placed on hold closes the audio stream, so the channel returns to Idle, and Air Time becomes zero. However, the same channel is used when the telephone retrieves the call from hold. |

**Table 87**
**Data output (cont'd.)**

| Parameter | Description |
|---|---|
| rxTsap | IP address and port VGMC receives RTP packets on. |
| txTsap | IP address and port VGMC sends RTP packets to. |

The following example shows command output on the ITG-P card.

```
VGMC> vgwShow
Chan ChanState DspMode Codec Tn Reg AirTime rxTsap txTsap
---- ---------- --------- ---------- ------ --- -------
-------------------- --------------------
0 Busy Voice G.711-20 0x0008 yes 11 10.1.1.2:5200
10.1.1.4:5200
1 Busy Voice G.711-20 0x0009 yes 11 10.1.1.2:5202
10.1.1.5:5200
2 Busy Voice G.711-20 0x000a yes 11 10.1.1.2:5204
10.1.1.6:5200
value = 3220 = 0xc94

VGMC> vgwShow "10.1.1.4"
value = 0 = 0x0
VGMC> Found on Card TN 002-00 , ELAN IP 47.11.214.52, TLAN IP
10.1.1.2
Chan ChanState DspMode Codec Tn Reg AirTime rxTsap txTsap
---- ---------- --------- ---------- ------ --- -------
-------------------- --------------------
0 Busy Voice G.711-20 0x0008 yes 68 10.1.1.2:5200
10.1.1.4:5200
```

In a NAT IP network environment, more than one IP Phone can map to a single IP address. The input is modified to allow the entry of the public port number for a specific telephone. If you enter no port number, the first entry found with the specified IP address on a VGMC is returned, as shown in the following example.

```
VGMC> vgwShow "47.11.215.136"
value = 0 = 0x0
-> Found on Card TN 005-00 , ELAN IP 47.11.216.174, TLAN IP
47.11.215.143, number of matches 2
Chan ChanState DspMode Codec Tn Reg AirTime rxTsap txTsap
---- ---------- --------- ---------- ------ --- -------
-------------------- --------------------
17 Busy Voice G.711-20 0x0505 yes 21 47.11.215.143:5234
47.11.215.136:2237
```

```
-> Found on Card TN 003-00 , ELAN IP 47.11.216.175, TLAN IP
47.11.215.146, number of matches 1
Chan ChanState DspMode Codec Tn Reg AirTime rxTsap txTsap
---- ---------- --------- ----------- ------ --- -------
-------------------- --------------------
1 Busy Voice G.711-20 0x0307 yes 21 47.11.215.145:5202
47.11.215.136:5200
```

When the IP address is found in the list of voice Gateway channels for a card other than where you entered the command, the voice gateway channel information for the first occurrence is returned plus a count of the number of times the IP address occurs in that card list. Multiple instances can occur when the customer network is configured so that multiple phones are behind a NAT device that shares the public IP address of the device.

If there is more than one match, you can log on to that card and enter the command without entering an IP address and port number to print all busy channels on the card. To quickly find a particular telephone, use the IPDN or DNIP commands in LD 117 to obtain the media stream public IP address and port number for the telephone, and then enter those as parameters for the vgwShow command.

## vgwShowAll

Syntax:

**`vgwShowAll`**

Print information about all gateway channels on a VGMC. This command is not available on the Signaling Server.

The following example shows command output.

```
VGMC> vgwShowAll
Chan ChanState DspMode Codec Tn Reg AirTime rxTsap txTsap
---- ---------- --------- ----------- ------ --- -------
-------------------- --------------------
0 Busy Voice G.711-20 0x0008 yes 7661 10.1.1.2:5200
10.1.1.4:5200
1 Busy Voice G.711-20 0x0009 yes 7661 10.1.1.2:5202
10.1.1.5:5200
2 Busy Voice G.711-20 0x000a yes 7661 10.1.1.2:5204
10.1.1.6:5200
3 Idle Closed n/a 0x000b yes 0 0.0.0.0:0000 0.0.0.0:0000
4 Idle Closed n/a 0x0048 yes 0 0.0.0.0:0000 0.0.0.0:0000
5 Idle Closed n/a 0x0049 yes 0 0.0.0.0:0000 0.0.0.0:0000
...  (one line printed per channel)
```

```
21 Idle Closed n/a 0x0149 yes 0 0.0.0.0:0000 0.0.0.0:0000
22 Idle Closed n/a 0x014a yes 0 0.0.0.0:0000 0.0.0.0:0000
23 Idle Closed n/a 0x014b yes 0 0.0.0.0:0000 0.0.0.0:0000
value = 98 = 0x62 = 'b'
```

### vgwTraceAllOff

Syntax:

**`vgwTraceAllOff`**

This command turns off the trace for all channels.

The following example shows command output.

```
-> vgwTraceAllOff
value = 0 = 0x0
```

### vgwTraceHelp

Syntax:

**`vgwTraceHelp`**

This command displays the CLIs associated with the trace tool and describes each CLI parameter.

### vgwTraceOff

Syntax:

**`vgwTraceOff chNum`**

This command turns off the trace for the specified channel.

The following table describes the command parameters.

**Table 88**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Channel number. |

### vgwTraceOn

Syntax:

**`vgwTraceOn chNum, vgw_trace_tool`**

This command initiates the audio message tracing.

The following table describes the command parameters.

**Table 89**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| chNum | – | Specify the voice Gateway or channel on which to perform ACP or audio message tracing. |
| vgw_trace_tool | – | Bitmask identifying the set of messages to trace for the specified channel.<br>0 = All voice Gateway Message Tracing<br>1 = A07 Message Tracing<br>2 = Audio Message Tracing<br>4 = Registration Message Tracing |

The following example shows command output.

```
-> vgwTraceOn 1,1
value = 0 = 0x0
```

## vgwTraceSetOutput

Syntax:

**`vgwTraceSetOutput trace_output, "file_pathname"`**

Assign the output destination for the trace tool.

The following table describes the command parameters.

**Table 90**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| trace_output | 1–4 | Trace output destination, where<br>1 = TTY<br>2 = SYSLOG<br>3 = File<br>4 = File and TTY |
| file_pathname | "string" | Specify the file to output to if trace_output = 3 or 4. Enclose the string in quotation marks.<br><br>You can enter the entire path name, or the file name only. If you enter only the file name, the default path is C:/file_name. |

The following example shows command output.

```
-> vgwTraceSetOutput 3, "trace.txt"
value = 0 = 0x0
```

## vtmUMSDownload

Syntax:

**`vtmUMSDownload "IPAddr"`**

This command initiates a request for a firmware download to the specific IP Phone.

The following table describes the command parameters.

**Table 91**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| file_pathname IPAddr | "string" | Public IP address of the IP Phone. Enclose the string in quotation marks. |

The following example shows command output.

```
-> vtmUMSDownload "147.11.215.136"
value = 0 = 0x0
-> 21/01/04 14:20:09 LOG0006 UMS: decision-bless
"147.11.215.136"
21/01/04 14:20:09 LOG0006 VTM: vtmTerminal Online
0x9b93da0:  set is already online
```

## vtmAPBSet

Syntax:

**`vtmAPBSet "IPAddr"`**

Refresh transducer APB values for a specific IP Phone. If no lossPlan is defined, the following default values are used:
HandsetRLR = 2
HandsetSLR = 11
HandsetSTMR = 18
HeadsetRLR = 0
HeadsetSLR = 11
HeadsetSTMR =18
HandsfreeRLR = 13
HandsfreeSLR = 16
HandsfreeSTMR = 22

The following table describes the command parameters.

**Table 92**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| IPAddr | "string" | Optional. Public IP address of the IP Phone. Enclose the string in quotation marks. |

The following example shows command output.

```
-> vtmAPBSet "147.11.215.135"
value = 0 = 0x0
```

### vxshell

Syntax:
**`vxshell`**

Access the OS shell from the the VGMC or the pdt> prompt on the Signaling Server.

The following example shows command output on the Signaling Server.

```
pdt> vxshell
->
```

The following example shows command output on the VGMC.

```
VGMC> vxshell
login: pdt2
password:
Welcome to the VxWorks Shell
WARNING: Data entry errors in this shell could cause loss of
service.
Use itgShell to return to the ITG shell.
value = 52688160 = 0x323f520
->
```

### wapListShow

Syntax:
**`wapListShow`**

Display all IP Phones that registered to the WAP (Web Application Proxy) module .

The following example shows command output.

```
> wapListShow
TN pWapSet appState SessWeb hForm defaultPostData Title
---- ---------- ---------- --------- --------
```

```
---------------------------------- ----------
60c5 0xbc4afb0 2 0xbdf2f20 0x0 uid=6113&cust=0&pni=0&hlo
c=&tn=24773&language=english (null)
```

### wapTraceOff

Syntax:
**`wapTraceOff "phoneIPAddress"`**

Turn off the hidden trace for a specific IP Phone.

The following example shows command output.

```
-> wapTraceOff "47.11.216.186"
WAP trace stopped for 47.11.216.186
```

### wapTraceOn

Syntax:
**`wapTraceOn "phoneIPAddress"`**

Turn on the hidden trace for a specific IP Phone (or more than one). The log is written into the /u/trace directory with the file name wap.log.

The following example shows command output.

```
-> wapTraceOn "47.11.216.186"
value = 789305530 = 0x2f0bd8ba
```

### wapTraceVerboseSet

Syntax:
**`wapTraceVerboseSet mode`**

This command allows WAP to print on-screen any document it receives from the application server. This command is useful to debug screen operation on single IP Phones.

The following table describes the command parameters.

**Table 93**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| mode | 0, 1 | 0 = Enables verbose mode for all IP Phones.<br>1 = Disables verbose mode for all IP Phones. |

The following example shows command output.

```
-> wapTraceVerboseSet 1
value = 1 = 0x1
-> wapTraceVerboseSet 0
value = 0 = 0x0
```

## wapWebFormShow

Syntax:

**`wapWebFormShow`**

Use wapListShow to obtain the Web handle displayed on the IP Phone, and then use this command to get a detailed view on this Web form.

The following example shows command output.

```
-> wapWebFormShow 0xadcec98
------------------WAP WebForm------------------
ACTION: pdperdir.cgi?fn=12
METHOD: get
POSTDATA name=Jicheng,Zhou&number=3904
FORMMENU: 0
ACTINPUT: 0
SIZEOFPO: 32
HCONTROL: 0x0
-------- Input Elements-----------
VAR: name TYPE: 0
VAL: Jicheng,Zhou
TXT: Enter Name:
MAXLEN: 24
POST: name=Jicheng,Zhou
HCONTR: 0xad9c784
PINPUT: 0xad939b8
VAR: number TYPE: 6
VAL: 3904
TXT: Enter Name:
MAXLEN: 31 POST: number=3904
HCONTR: 0x0
PINPUT: 0xadcea
```

## webClientShow

Syntax:

**`webClientShow`**

Print statistics for the web client.

The following example shows command output.

```
-> webClientShow
Failed on getting web client handle ...........  1
```

## XspySetLevel

Syntax:

**`XspySetLevel key, filterLevel`**

Xspy is a generic function in the Telogy code through which error and information messages are printed. This command applies only to the VGMCs.

This command allows you to configure what prints from the various components of the Telogy code and is global to all DSP channels. For example, you can enable printing of the commands sent between the DIM module and the DSP. These commands can display information about gain settings, echo canceller control, and other parameters sent to the DSP when a call is set up.

The following table describes the command parameters.

**Table 94**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| key | 0, 1 | Specify a component of the Telogy code, where<br>0 = All<br>1 = Root<br><br>The recommended value is 0. |
| filterLevel | 0 - 6 | Specify the minimum level of messaging to print, where<br>0 = General information<br>1 = Function entry<br>2 = Normal event (items such as STATUS REQ/RESP polling do not print)<br>3 = Minor unexpected event<br>4 = Major unexpected event<br>5 = Fatal error<br>6 = Spy trace off (turns off printing)<br><br>The recommended value is 0 or 2. |

> ***Note:*** If you enter the XspySetLevel command without parameters, the parameters default to XspySetLevel 0,0.

The following example shows the output when a call is made.

```
-> XspySetLevel 0,0
value = 0 = 0x0
```

ITG-P card is idle:

```
OCT 23 11:29:27 tMVX_XSPY: Info 0083725626 - DIM: 0:0,
Tx='STATUS_REQUEST'(8)
OCT 23 11:29:27 tMVX_XSPY: Info 0083725626 - DIM: 0:*, Rx
'TONE_DETECT'(101) Len= 5
OCT 23 11:29:27 tMVX_XSPY: Info 0083725818 - DIM: 1:0,
Tx='STATUS_REQUEST'(8)
OCT 23 11:29:27 tMVX_XSPY: Info 0083725818 - DIM: 1:*, Rx
'TONE_DETECT'(101) Len= 5
OCT 23 11:29:28 tMVX_XSPY: Info 0083726078 - DIM: 2:0,
Tx='STATUS_REQUEST'(8)
OCT 23 11:29:28 tMVX_XSPY: Info 0083726078 - DIM: 2:*, Rx
'TONE_DETECT'(101) Len= 5
OCT 23 11:29:33 tMVX_XSPY: Info 0083728492 - DIM: 3:0,
Tx='STATUS_REQUEST'(8)
OCT 23 11:29:33 tMVX_XSPY: Info 0083728492 - DIM: 3:*, Rx
'TONE_DETECT'(101) Len= 5
OCT 23 11:29:33 tMVX_XSPY: Info 0083728740 - DIM: 4:0,
Tx='STATUS_REQUEST'(8)
OCT 23 11:29:33 tMVX_XSPY: Info 0083728740 - DIM: 4:*, Rx
'TONE_DETECT'(101) Len= 5
OCT 23 11:29:34 tMVX_XSPY: Info 0083728990 - DIM: 5:0,
Tx='STATUS_REQUEST'(8)
OCT 23 11:29:34 tMVX_XSPY: Info 0083728990 - DIM: 5:*, Rx
'TONE_DETECT'(101) Len= 5
OCT 23 11:29:34 tMVX_XSPY: Info 0083729238 - DIM: 6:0,
Tx='STATUS_REQUEST'(8)
OCT 23 11:29:34 tMVX_XSPY: Info 0083729238 - DIM: 6:*, Rx
'TONE_DETECT'(101) Len= 5
OCT 23 11:29:35 tMVX_XSPY: Info 0083729488 - DIM: 7:0,
Tx='STATUS_REQUEST'(8)
OCT 23 11:29:35 tMVX_XSPY: Info 0083729488 - DIM: 7:*, Rx
'TONE_DETECT'(101) Len= 5
```

IP to TDM call:

```
->
OCT 23 11:26:22 tMVX_XSPY: Info 0083633318 - DIM: 0:0,
dim_open_channel.  TCID = 0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633320 - DIM: 0:0,Tx='O
```

```
PEN_CHANNEL'(13):tx=9,rx=9,TxVIF=640,RxVIF=640,VAD=0,Co
mp=1,NomDel=60,MaxDel=120,IdleNois=-6500,EcTail=1024,
OCT 23 11:26:22 tMVX_XSPY: Info 0083633320 - ----VadThrsh=-
17,TxG=0,RxG=0,TxInG=0,Encap=3,AdPOut=1,Ts=1,Resamp=0,P
ktIO=0,PktSIO_Ts=0,EcCfg=0x2,NlpFixCfg=0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633320 - DIM: Tcid 0.
Swo Idle => Swo Initial
OCT 23 11:26:22 tMVX_XSPY: Info 0083633322 - DIM: 0:0,
Request channel poll period 10 msecs
OCT 23 11:26:22 tMVX_XSPY: Info 0083633322 - DIM: 0:0, DSP
bpm 0, Channel bpm 0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633322 - DIM: 0, Req
poll period set to 4 msecs
OCT 23 11:26:22 tMVX_XSPY: Info 0083633322 - DIM: 0:0,
Request channel poll period 5 msecs
OCT 23 11:26:22 tMVX_XSPY: Info 0083633322 - DIM: 0:0, DSP
bpm 10, Channel bpm 10
OCT 23 11:26:22 tMVX_XSPY: Info 0083633322 - DIM: 0, Req
poll period set to 4 msecs
OCT 23 11:26:22 tMVX_XSPY: Info 0083633324 - DIM: 0:0,
Tx='DTMF_REL_ENABLE'(40)
OCT 23 11:26:22 tMVX_XSPY: Info 0083633326 - DIM: 0:0,
Tx='DTMF_MODE'(2) detect mode=0x0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633328 - DIM: 0:0,
Tx='VOICE_MODE'(11) ttu_enable=0, start_mark=0, t_tone=0,
t_s1=0, t_s1s2=0, seq_S1=0, seq_S2=0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633330 - DIM: 0:0,
Tx='EC_CONTROL'(3) Len=12 valid_bitfield=0x1c,tail_len=0
,noise_lev=0, config_bits=0x40
OCT 23 11:26:22 tMVX_XSPY: Info 0083633330 - DIM: 0:0, ----
config_bits=0x40, nlp_aggress=0, comfort_noise_cfg=0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633330 - DIM: 0:0, ----
Valid Bitfield:  EC_TAIL_VALID=0,EC_NLEVEL_VALID=0,
CONFIG_VALID=1
OCT 23 11:26:22 tMVX_XSPY: Info 0083633330 - DIM: 0:0, ----
Valid Bitfield:  EC_NLPAGG_VALID=1, EC_CN_CFG_VALID=1
OCT 23 11:26:22 tMVX_XSPY: Info 0083633330 - DIM: 0:0,
---- config EC_DISABLE=0, EC_FREEZE=0,EC_NLP_DISABLE=0,
EC_AUTO_UPD_DISABLE=0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633330 - DIM: 0:0,
---- config EC_MAG_FREEZE=0, EC_SRCH_FREEZE_DISABLE=1,
EC_NLP_CN_FIXED=0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633330 - DIM: 0:0,
---- config EC_4W_DETECT_DISABLE=0, EC_TEST_BIT_0=0,
EC_TEST_BIT_1=0, EC_TEST_BIT_2=0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633330 - DIM:
0:0, ---- config EC_TEST_BIT_3=0, EC_TEST_BIT_4=0,
EC_NLP_FORCE_ENABLE=0, EC_NLP_NORM_LEVEL=0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633332 - DIM: 0:0,
Tx='ENABLE_TONE_DET'(4) tone=V.21 TONE
OCT 23 11:26:22 tMVX_XSPY: Info 0083633334 - DIM: 0:0,
Tx='ENABLE_TONE_DET'(4) tone=V.25 TONE
OCT 23 11:26:22 tMVX_XSPY: Info 0083633336 - DIM: 0:0,
Tx='SET_RX_GAIN'(43) gain=0x0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633338 - DIM: 0:0,
Tx='SET_TX_GAIN'(42) gain=+0, inGain=+0
OCT 23 11:26:22 tMVX_XSPY: Info 0083633350 - DIM: 0:0,
Tx='STATUS_REQUEST'(8)
OCT 23 11:26:22 tMVX_XSPY: Info 0083633350 - DIM: 0:*, Rx
'TONE_DETECT'(101) Len= 5
OCT 23 11:26:22 tMVX_XSPY: Info 0083633386 - DIM: Tcid 0.
Swo Initial => Swo Ready
OCT 23 11:26:23 tMVX_XSPY: Info 0083633454 - DIM: 3:0,
Tx='STATUS_REQUEST'(8)
OCT 23 11:26:23 tMVX_XSPY: Info 0083633454 - DIM: 3:*, Rx
'TONE_DETECT'(101) Len= 5
```

Releasing the call:

```
->
OCT 23 11:29:17 tMVX_XSPY: Info 0083720618 - DIM: 0:0, stat
req cmd=65 clear=0
OCT 23 11:29:17 tMVX_XSPY: Info 0083720618 - DIM: 0:0, stat
req cmd=66 clear=0
OCT 23 11:29:17 tMVX_XSPY: Info 0083720618 - DIM: 0:0, stat
req cmd=67 clear=0
OCT 23 11:29:17 tMVX_XSPY: Info 0083720620 - DIM: 0:0,
Tx='GET_VP_STAT'(65)
OCT 23 11:29:17 tMVX_XSPY: Info 0083720620 - DIM: 0:0, Rx
'GET_VP_STAT'(65) Len=17
OCT 23 11:29:17 tMVX_XSPY: Info 0083720620 - DIM: 0:0 -
MGB_DM_GET_VP_DELAY
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, avg_playout_delay =
56
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, lost_packet_count = 0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, replay_packet_count
= 0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, idle_packet_count = 0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, dropped_packet_count
= 0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, rx_packet_count =
17446
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, avg_frame_jitter = 0
OCT 23 11:29:17 tMVX_XSPY: Info 0083720622 - DIM: 0:0,
Tx='GET_ERROR_STAT'(66) clear=0x0
OCT 23 11:29:17 tMVX_XSPY: Info 0083720622 - DIM: 0:0, Rx
'GET_ERROR_STAT'(66) Len=15
OCT 23 11:29:17 tMVX_XSPY: Info 0083720622 - DIM: 0:0 -
MGB_DM_GET_ERROR_STAT
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, invalid_header_count
= 0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, to_micro_overflow_co
unt = 0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, lost_enh_packet_coun
t = 0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, no_core_packet_count
= 0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, pkt_lost_by_network
= 0
OCT 23 11:29:17 tMVX_XSPY: Info 0083720624 - DIM: 0:0,
Tx='GET_RXTX_STAT'(67) clear=0x0
OCT 23 11:29:17 tMVX_XSPY: Info 0083720624 - DIM: 0:0, Rx
'GET_RXTX_STAT'(67) Len=29
OCT 23 11:29:17 tMVX_XSPY: Info 0083720624 - DIM: 0:0 -
MGB_DM_GET_RXTX_STAT
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, rx_packet_count =
17446
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, tx_packet_count =
17456
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, silence_packet_count
= 0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, min_jitter = 5
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, max_jitter = 15
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, rtp_average_jitter =
0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, tx_grant_sync_drop =
0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, tx_octets = 1396480
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, rx_octets = 1395680
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, aal2_cod_prof_chgs =
0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, dtmf_tx_octets = 0
OCT 23 11:29:17 tMVX_DIM: Info TCID 0, dtmf_rx_octets = 0
OCT 23 11:29:17 tMVX_XSPY: Info 0083720626 - DIM: 0:0,
Tx='CLOSE_CHAN'(14)
OCT 23 11:29:17 tMVX_XSPY: Info 0083720626 - DIM: 0:0,
Request channel poll period 4 msecs
OCT 23 11:29:17 tMVX_XSPY: Info 0083720626 - DIM: 0:0, DSP
bpm 0, Channel bpm 0
OCT 23 11:29:17 tMVX_XSPY: Info 0083720626 - DIM: 0, Req
poll period set to 4 msecs
```

```
OCT 23 11:29:17 tMVX_XSPY: Info 0083720626 - DIM: Tcid 0.
Swo Ready => Swo Idle
OCT 23 11:29:17 tMVX_XSPY: Info 0083720626 - DIM 0,0:  DSP
got packet for a closed or disassoc channel (len=100,
state=1)
OCT 23 11:29:17 tMVX_XSPY: Info 0083720816 - DIM: 1:0,
Tx='STATUS_REQUEST'(8)
OCT 23 11:29:17 tMVX_XSPY: Info 0083720816 - DIM: 1:*, Rx
'TONE_DETECT'(101) Len= 5
```

# MGC command reference

The commands and variables in this section are listed in alphabetical order.

## dbhwshow

Syntax:
**dbhwshow**

Display the model and revision numbers for installed DBs.

The following example shows command output.

```
oam> dbhwshow
DB1 model and revision no:  NTDW62AA
```

## diskshow

Syntax:
**diskshow**

Display the total, used, and available disk space on the internal flash card.

The following example shows command output.

```
oam> diskshow
Partition /p Total:  49.00MB Used:  37.26MB Avail:  11.74MB
Partition /p1 Total:  48.98MB Used:  37.26MB Avail:
11.73MB
Partition /e Total:  12.00MB Used:  8.75MB Avail:  3.25MB
Partition /u Total:  12.00MB Used:  0.77MB Avail:  11.23MB
```

## displayshow

Syntax:
**displayshow**

Display the information currently showing on the four-character faceplate display. If the display cycles through multiple messages, all messages are shown.

The following example shows command output.

```
oam> displayshow

020 0
```

## dspnumshow

Syntax:
**`dspnumshow status`**

Display the number of DSP channels for each DSP DB in the specified mode.

The following table describes the command parameters.

**Table 95**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| status | IDLE, UNEQ, DSBL, BUSY | Optional. Specify the mode for which to display the number of DSP channels, where:<br>IDLE = idle<br>UNEQ = unequipped<br>DSBL = disabled<br>BUSY = busy |

The following example shows command output.

```
oam> dspnumshow idle
Card :  11 32 Channels are:  Idle
oam> dspnumshow uneq
Card :  11 0 Channels are:  Uneq
oam> dspnumshow dsbl
Card :  11 0 Channels are:  Dsbl
oam> dspnumshow busy
Card :  11 0 Channels are:  Busy
```

## dspchanstateshow

Syntax:
**`dspchanstateshow`**

List the state (busy, idle, disabled, or unequipped) for all channels on the DSP DBs.

The following example shows command output.

```
oam> dspchanstateshow
Card No.:  11
Channels:  0 1 2 3 4 5 6 7
Idle Idle Idle Idle Idle Idle Idle Idle
Channels:  8 9 10 11 12 13 14 15
Idle Idle Idle Idle Idle Idle Idle Idle
Channels:  16 17 18 19 20 21 22 23
Idle Idle Idle Idle Idle Idle Idle Idle
Channels:  24 25 26 27 28 29 30 31
Idle Idle Idle Idle Idle Idle Idle Idle
```

## dsphwcheck

Syntax:
**`dsphwcheck`**

Perform a basic DSP hardware diagnostic check, testing for any hardware failures.

The following example shows command output.

```
oam> dsphwcheck
Daughterboard 1 :  NTDW62AA (32 Channel)
Initialization /self - test result :  PASS
Version Check :  4
Checksum :  1d03
Device Type :  M82520
G.711 VBE : PASS
Daughterboard 2 :  Not Installed
```

## dsplooptest

Syntax:
**`dsplooptest card1 channel1 card2 channel2`**

Perform a DSP loopback test on idle channels.

The following table describes the command parameters.

**Table 96**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| card1, card2 | – | Cards on which to test channels. Disable the cards in Call Server LD 32 before using the dsplooptest command. |
| channel1, channel2 | – | Channels to test. Ensure that the channels reside on the same physical DSP device. |

The following example shows command output.

```
oam> dsplooptest 11 0 11 1

!!! This command will cause service interuption !!!

!!! Please go to CS LD32 disable card 11 unit 0 and card 11
unit 1 before performing this test !!!

!!! This command uses card 10 unit 0 and card 10 unit 1 as
well. Disabling these units is recommended !!!

Do you want to proceed? (y/n/[a]bort) : y

Channel 1 Channel 2 Status
---------- ---------- --------
11 0 11 1 PASS
```

## ethportmirror

Syntax:
**`ethportmirror <to port><from port>`**

Capture traffic from a specified port to a mirrored port.

Use port mirroring to connect LAN analyzer equipment to the Layer 2 switch and capture LAN traffic on external LAN ports (such as Layer 2 TLAN/ELAN and 100BaseT Media Gateways). You can also use port mirroring to capture the Signaling or proprietary message traffic (Mindspeed Tone and Conf Module, Expansion Boards, and VoIP Daughter Boards) between internal Layer 2 components.

**Table 97**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| to port | 2–7 | Port used for mirroring.<br><br>Disable this port by using the ethportdisable command before you can use this port for port mirroring. Use a port that is not in use (no TLAN or ELAN traffic). |
| from port | 0–9 | Port to mirror |

The following example shows command output.

```
ldb> ethportdisable 4
Disable port 4 ([y]/n) :
ldb> ethportmirror 4 3
Mirror port 4 to port 3 ([y]/n) :
ldb> ethportshow

Port 0 = DB2 Internal
Port 1 = DB1 Internal
Port 2 = 1(ELAN) Back plane
Port 3 = 2(TLAN) Back plane
Port 4 = CE Top face plate connector for CPPM only
Port 5 = CT Second from top face plate connector for CPPM
only
Port 6 = 1E Second from bottom face plate connector
Port 7 = 2T Bottom face plate connector
Port 8 = MSP Internal
Port 9 = CSP Internal
0x10=0x213b -Port 0 sends to 0 1 3 4 5 8
0x11=0x213b -Port 1 sends to 0 1 3 4 5 8
0x12=0x1200 -Port 2 sends to 9
0x13=0x2133 -Port 3 sends to 0 1 4 5 8
0x14=0x0000 -Port 4 sends to
0x15=0x211b -Port 5 sends to 0 1 3 4 8
0x16=0x1200 -Port 6 sends to 9
0x17=0x2123 -Port 7 sends to 0 1 5 8
0x18=0x203b -Port 8 sends to 0 1 3 4 5
0x19=0x1040 -Port 9 sends to 6
0x10=0x0001 -Port 0 receives from 0
0x11=0x0002 -Port 1 receives from 1
0x12=0x0004 -Port 2 receives from 2
0x13=0x8018 -Port 3 receives from 3 Is mirrored on port 4
0x14=0x0010 -Port 4 receives from 4
0x15=0x0020 -Port 5 receives from 5
0x16=0x0040 -Port 6 receives from 6
```

```
0x17=0x0080 -Port 7 receives from 7
0x18=0x0100 -Port 8 receives from 8
0x19=0x0200 -Port 9 receives from 9
Current vlan table number is 1
Dual homing is disabled.
Carrier detected on ports 3 6
Port 2 is running at 10 Mbps half duplex.
Port 3 is running at 100 Mbps half duplex.
Port 4 is running at 10 Mbps half duplex.
Port 5 is running at 10 Mbps half duplex.
Port 6 is running at 100 Mbps full duplex.
Port 7 is running at 10 Mbps half duplex.
Port: Designator
====== ============
CE: CPPMELAN
1E: FPELAN
E: BPELAN
CT: CPPMTLAN
2T: FPTLAN
T: BPTLAN
```

## ethportshow

Syntax:
**`ethportshow`**

Display the Ethernet port settings for the external and internal interfaces. The output includes autonegotiation settings, duplex, port speed and port mirroring status.

The following example shows command output.

```
oam> ethportshow
Port 0 = DB2 Internal
Port 1 = DB1 Internal
Port 2 = 1(ELAN) Back plane
Port 3 = 2(TLAN) Back plane
Port 4 = CE Top face plate connector for CPPM only
Port 5 = CT Second from top face plate connector for CPPM
only
Port 6 = 1E Second from bottom face plate connector
Port 7 = 2T Bottom face plate connector
Port 8 = MSP Internal
Port 9 = CSP Internal

0x10=0x21a3 -Port 0 sends to 0 1 5 7 8
0x11=0x21a3 -Port 1 sends to 0 1 5 7 8
0x12=0x1210 -Port 2 sends to 4 9
```

```
0x13=0x2123 -Port 3 sends to 0 1 5 8
0x14=0x1240 -Port 4 sends to 6 9
0x15=0x2183 -Port 5 sends to 0 1 7 8
0x16=0x1210 -Port 6 sends to 4 9
0x17=0x2123 -Port 7 sends to 0 1 5 8
0x18=0x20a3 -Port 8 sends to 0 1 5 7
0x19=0x1050 -Port 9 sends to 4 6

0x10=0x0001 -Port 0 receives from 0
0x11=0x0002 -Port 1 receives from 1
0x12=0x0044 -Port 2 receives from 2 6
0x13=0x0088 -Port 3 receives from 3 7
0x14=0x0010 -Port 4 receives from 4
0x15=0x0020 -Port 5 receives from 5
0x16=0x0044 -Port 6 receives from 2 6
0x17=0x0088 -Port 7 receives from 3 7
0x18=0x0100 -Port 8 receives from 8
0x19=0x0200 -Port 9 receives from 9

Current vlan table number is 0

Dual homing is enabled.

Carrier detected on ports 6 7

vlanPHYResolvedPrint:port 2 cmd 851 retval 50
Auto-negotiation is not complete for port 2!
Port 2 is running at 10 Mbps half duplex.
vlanPHYResolvedPrint:port 3 cmd 871 retval 50
Auto-negotiation is not complete for port 3!
Port 3 is running at 10 Mbps half duplex.
vlanPHYResolvedPrint:port 4 cmd 891 retval 10
Auto-negotiation is not complete for port 4!
Port 4 is running at 10 Mbps half duplex.
vlanPHYResolvedPrint:port 5 cmd 8b1 retval 10
Auto-negotiation is not complete for port 5!
Port 5 is running at 10 Mbps half duplex.
vlanPHYResolvedPrint:port 6 cmd 8d1 retval 7c40
Port 6 is running at 100 Mbps full duplex.
vlanPHYResolvedPrint:port 7 cmd 8f1 retval 7c40
Port 7 is running at 100 Mbps full duplex.

Port:  Designator
====== ============
CE: CE
1E: 1E
E: E
```

```
CT: CT
2T: 2T
T: T
```

## ethspeedshow

Syntax:

**`ethspeedshow`**

Display the port speed and duplex setting of the embedded Ethernet switch currenlty running.

The following example shows command output.

```
oam> ethspeedshow
vlanPHYResolvedPrint:port 2 cmd 851 retval 50
Auto-negotiation is not complete for port 2!
Port 2 is running at 10 Mbps half duplex.
vlanPHYResolvedPrint:port 3 cmd 871 retval 50
Auto-negotiation is not complete for port 3!
Port 3 is running at 10 Mbps half duplex.
vlanPHYResolvedPrint:port 4 cmd 891 retval 10
Auto-negotiation is not complete for port 4!
Port 4 is running at 10 Mbps half duplex.
vlanPHYResolvedPrint:port 5 cmd 8b1 retval 10
Auto-negotiation is not complete for port 5!
Port 5 is running at 10 Mbps half duplex.
vlanPHYResolvedPrint:port 6 cmd 8d1 retval 6c40
Port 6 is running at 100 Mbps full duplex.
vlanPHYResolvedPrint:port 7 cmd 8f1 retval 6c40
Port 7 is running at 100 Mbps full duplex.
```

## isecIkeShowPAll

Syntax:

**`isecIkeShowPAll`**

Display all protection suites (inbound and outbound IPsecSecurity association pairs).

The following example shows command output.

```
oam> isecIkeShowPAll
IPSec has not been initialized
```

## isecIpsecShowIf

Syntax:

**`isecIpsecShowIf`**

The following example shows command output.

```
oam> isecIpsecShowIf
IPSec has not been intialized
```

## macshow

Syntax:
**`macshow`**

Display all MAC addresses associated with the Ethernet ports (both internal and external) on the embedded Ethernet switch.

The following example shows command output.

```
oam> macshow
ELAN : 00:13:65:ff:ed:a2
TLAN : 00:13:65:ff:ed:a3
DB1 : 00:13:65:ff:ed:73
DB2 : NOT AVAILABLE
```

## memshow

Syntax:
**`memshow`**

Display the total, used, and available RAM memory on the card.

The following example shows command output.

```
oam> memshow
Total:  121.92MB Used:  60.26MB Avail:  61.66MB
```

## mgcdbshow

Syntax:
**`mgcdbshow card_number`**

Display information about the DSP DB.

The following table describes the command parameters.

**Table 98
Command parameters**

| Parameter | Value | Description |
|---|---|---|
| card_number | 0, 11, 12, 13 | Card number |

The following example shows command output.

```
oam> mgcdbshow 11
Initialized :  1
Type :  EXUT
TLAN DSP IP : 10.10.10.65
Card TN : 20 0 11
Card State :  ENBL
Uptime :  2 days, 19 hours, 49 mins, 28 secs (244168 secs)
Codecs :  G711Ulaw(default), G711Alaw, G729A, G711CC,
T38FAX
```

## mgcinfoshow

Syntax:

**`mgcinfoshow`**

Display basic information about the MGC, including IP addresses, uptime, registration status, and superloop information.

The following example shows command output.

```
oam> mgcinfoshow
Registration Status:  Registered on 47.11.214.87
UpTime:  2 days, 19 hours, 27 minutes, 43 seconds

MGC Hostname:  justinsnewhost
MGC ELAN IP Address:  47.11.214.83 MGC
Subnet Mask:  255.255.255.0
MGC Gateway Address:  47.11.214.1

MGC TLAN IP Address:  10.10.10.97
MGC Subnet Mask:  255.255.254.0
MGC Default GW: 10.10.10.1

Primary CS Hostname:  biglabCS
Primary CS IP Address:  47.11.214.87

DB1 NT code:  NTDW62AA
DB1 TLAN IP Address:  10.10.10.65
DB1 TN: 020 0 11

Media Gateway Information
Superloop Shelf:  020 0
TDS Loop:  100
CONF Loop:  101

Active TLAN port:  Port 7 = 2T Bottom face plate connector
TLAN port designator:  2T
TLAN set to auto negotiate.
```

```
Active ELAN port:  Port 6 = 1E Second from bottom face plate
connector
ELAN port designator:  1E
ELAN set to auto negotiate.
ELAN security Disabled
```

## mspversionshow

Syntax:
**`mspversionshow`**

Display MSP Device type, ARM code, Voice DSP Revision, and T.38 version.

The following example shows command output.

```
oam> mspversionshow
MSP Device 0 (TnC):
Device Type :  M82515
Device Rev.  :  REV_B
ARM Code :  v5_07
Voice DSP : C64V_5_17_2
Fax Version :  T38DDP_VER_5_1_3
MSP Device 1 (DB1):  (32 Channel)
Device Type :  M82520
Device Rev.  :  REV_E
ARM Code :  Branch_5_07
Rev E - HP Voice DSP : C64V_5_17_2
Fax Version :  T38DDP_VER_5_1_3
MSP Device 2 (DB2):  Not installed
```

## ommshow

Syntax:
**`ommshow`**

Print the current OM data to the console.

The following example shows command output.

```
oam> ommshow
collection_time :  12/14/2006 18:31

- VGW Call Status (Current hourly period) -
ChanAud_Setup:  0
ChanJitter_Avg:  0.00
ChanJitter_Max:  0
```

```
ChanPkt_Lost:  0.00
ChanLatency_Avg:  0.00
ChanVoice_Time:  0 mins 0 secs

<Card 0>
Unit Call Attempts Success Fails
---- -------------------------------------------
00 0 0 0
01 0 0 0
02 0 0 0
03 0 0 0
04 0 0 0
05 0 0 0
06 0 0 0
07 0 0 0
08 0 0 0
09 0 0 0
10 0 0 0
11 0 0 0
12 0 0 0
13 0 0 0
14 0 0 0
15 0 0 0
16 0 0 0
17 0 0 0
18 0 0 0
19 0 0 0
20 0 0 0
21 0 0 0
22 0 0 0
23 0 0 0
24 0 0 0
25 0 0 0
26 0 0 0
27 0 0 0
28 0 0 0
29 0 0 0
30 0 0 0
31 0 0 0

<Card 11>
Unit Call Attempts Success Fails
---- -------------------------------------------
00 0 0 0
01 0 0 0
02 0 0 0
03 0 0 0
```

```
04 0 0 0
05 0 0 0
06 0 0 0
07 0 0 0
08 0 0 0
09 0 0 0
10 0 0 0
11 0 0 0
12 0 0 0
13 0 0 0
14 0 0 0
15 0 0 0
16 0 0 0
17 0 0 0
18 0 0 0
19 0 0 0
20 0 0 0
21 0 0 0
22 0 0 0
23 0 0 0
24 0 0 0
25 0 0 0
26 0 0 0
27 0 0 0
28 0 0 0
29 0 0 0
30 0 0 0
31 0 0 0

<Card 12>
Unit Call Attempts Success Fails
---- ------------------------------------------
00 0 0 0
01 0 0 0
02 0 0 0
03 0 0 0
04 0 0 0
05 0 0 0
06 0 0 0
07 0 0 0
08 0 0 0
09 0 0 0
10 0 0 0
11 0 0 0
12 0 0 0
13 0 0 0
14 0 0 0
```

```
15 0 0 0
16 0 0 0
17 0 0 0
18 0 0 0
19 0 0 0
20 0 0 0
21 0 0 0
22 0 0 0
23 0 0 0
24 0 0 0
25 0 0 0
26 0 0 0
27 0 0 0
28 0 0 0
29 0 0 0
30 0 0 0
31 0 0 0

<Card 13>
Unit Call Attempts Success Fails
---- ------------------------------------------
00 0 0 0
01 0 0 0
02 0 0 0
03 0 0 0
04 0 0 0
05 0 0 0
06 0 0 0
07 0 0 0
08 0 0 0
09 0 0 0
10 0 0 0
11 0 0 0
12 0 0 0
13 0 0 0
14 0 0 0
15 0 0 0
16 0 0 0
17 0 0 0
18 0 0 0
19 0 0 0
20 0 0 0
21 0 0 0
22 0 0 0
23 0 0 0
24 0 0 0
25 0 0 0
```

```
26 0 0 0
27 0 0 0
28 0 0 0
29 0 0 0
30 0 0 0
31 0 0 0

- VGW Call Status (Since the system is up) -
Total VGW Call Attempts:  0
Total VGW Call Success:  0
Total VGW Call Failures:  0
Size of buffer 7281
```

## rmonstatreset

Syntax:

**`rmonstatreset`**

Reset all RMON statistics counters for a port on the embedded Ethernet switch.

The following example shows command output.

```
oam> rmonstatreset 1
Statistics counters have been reset for port 1
Port 1 = DB1 Internal
```

## rmonstatresetall

Syntax:

**`rmonstatresetall`**

Reset all RMON statistics counters for all ports on the embedded Ethernet switch.

The following example shows command output.

```
oam> rmonstatresetall
Statistics counters have been reset for port 0
Port 0 = DB2 Internal
Statistics counters have been reset for port 1
Port 1 = DB1 Internal
Statistics counters have been reset for port 2
Port 2 = 1(ELAN) Back plane
Statistics counters have been reset for port 3
Port 3 = 2(TLAN) Back plane
Statistics counters have been reset for port 4
Port 4 = CE Top face plate connector for CPPM only
```

```
Statistics counters have been reset for port 5
Port 5 = CT Second from top face plate connector for CPPM
only
Statistics counters have been reset for port 6
Port 6 = 1E Second from bottom face plate connector
Statistics counters have been reset for port 7
Port 7 = 2T Bottom face plate connector
Statistics counters have been reset for port 8
Port 8 = MSP Internal
Statistics counters have been reset for port 9
Port 9 = CSP Internal
```

## rmonstatshow

Syntax:

**`rmonstatshow port`**

Display the RMON statistics collected by the embedded Ethernet switch for the specified port.

The following table describes the command parameters.

**Table 99**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| port | 0–9 | Port number |

The following example shows command output.

```
oam> rmonstatshow 3
Statistics for port 3:
Port 3 = 2(TLAN) Back plane
Total good frames received = 793
Total good frames transmitted = 23991

Ingress statistics for port 3.
INUNICASTS = 26
INBROADCASTS = 677
INPAUSE = 0
INMUILICASTS = 90
INFCSERR = 0
ALIGNERR = 0
INGOODOCTETS = 68525
INBADOCTETS = 0
UNDERSIZE = 0
FRAGMENTS = 0
IN64OCTETS = 625
IN127OCTETS = 29
```

```
IN255OCTETS = 130
IN511OCTETS = 7
IN1023OCTETS = 2
INMAXOCTETS = 0
JABBER = 0
OVERSIZE = 0
INDISCARDS = 0
INFILTERED = 0

Egress statistics for port 3.
OUTUNICASTS = 332
OUTBROADCASTS = 3436858
OUTPAUSE = 0
OUTMUILICASTS = 322353
OUTFCSERR = 0
OUTOCTETS = 331886433
OUT64OCTETS = 2861144
OUT127OCTETS = 151412
OUT255OCTETS = 718156
OUT511OCTETS = 27747
OUT1023OCTETS = 1083
OUTMAXOCTETS = 1
COLLISIONS = 0
LATE = 0
EXECESSIVE = 0
multiple = 0
SINGLE = 0
DEFERRED = 0
OUTFILTERED = 0
OUTDISCARDS = 0
```

## rmonstatshowall

Syntax:

**`rmonstatshowall`**

Display the RMON statistics collected by the embedded Ethernet switch for all ports.

The following example shows command output.

```
am> rmonstatshowall
=============================== Port 0
===============================

Statistics for port 0:
Port 0 = DB2 Internal
Total good frames received = 0
Total good frames transmitted = 0
vlanStatPrint:  writing data d000 register = 1b address = 1d

Ingress statistics for port 0.
INUNICASTS = 0
INBROADCASTS = 0
INPAUSE = 0
INMUILICASTS = 0
INFCSERR = 0
ALIGNERR = 0
INGOODOCTETS = 0
INBADOCTETS = 0
UNDERSIZE = 0
FRAGMENTS = 0
IN64OCTETS = 0
IN127OCTETS = 0
IN255OCTETS = 0
IN511OCTETS = 0
IN1023OCTETS = 0
INMAXOCTETS = 0
JABBER = 0
OVERSIZE = 0
INDISCARDS = 0
INFILTERED = 0

Egress statistics for port 0.
OUTUNICASTS = 0
OUTBROADCASTS = 0
OUTPAUSE = 0
OUTMUILICASTS = 0
OUTFCSERR = 0
OUTOCTETS = 0
OUT64OCTETS = 0
OUT127OCTETS = 0
OUT255OCTETS = 0
OUT511OCTETS = 0
OUT1023OCTETS = 0
OUTMAXOCTETS = 0
COLLISIONS = 0
LATE = 0
EXECESSIVE = 0
multiple = 0
SINGLE = 0
DEFERRED = 0
OUTFILTERED = 72563
OUTDISCARDS = 0
```

```
=============================== Port 1 ==================
=============
Statistics for port 1:
Port 1 = DB1 Internal
Total good frames received = 1199
Total good frames transmitted = 8228
vlanStatPrint:  writing data d001 register = 1b address = 1d

Ingress statistics for port 1.
INUNICASTS = 1199
INBROADCASTS = 0
INPAUSE = 0
INMUILICASTS = 0
INFCSERR = 0
ALIGNERR = 0
INGOODOCTETS = 76762
INBADOCTETS = 0
UNDERSIZE = 0
FRAGMENTS = 0
IN64OCTETS = 1198
IN127OCTETS = 1
IN255OCTETS = 0
IN511OCTETS = 0
IN1023OCTETS = 0
INMAXOCTETS = 0
JABBER = 0
OVERSIZE = 0
INDISCARDS = 0
INFILTERED = 0

Egress statistics for port 1.
OUTUNICASTS = 1221
OUTBROADCASTS = 72539
OUTPAUSE = 0
OUTMUILICASTS = 4
OUTFCSERR = 0
OUTOCTETS = 6180668
OUT64OCTETS = 72636
OUT127OCTETS = 7
OUT255OCTETS = 28
OUT511OCTETS = 3
OUT1023OCTETS = 1
OUTMAXOCTETS = 1089
COLLISIONS = 0
LATE = 0
EXECESSIVE = 0
multiple = 0
```

```
SINGLE = 0
DEFERRED = 0
OUTFILTERED = 0
OUTDISCARDS = 0

=============================== Port 2 ==================
=============

Statistics for port 2:
Port 2 = 1(ELAN) Back plane
Total good frames received = 0
Total good frames transmitted = 0
vlanStatPrint:  writing data d002 register = 1b address = 1d

Ingress statistics for port 2.
INUNICASTS = 0
INBROADCASTS = 0
INPAUSE = 0
INMUILICASTS = 0
INFCSERR = 0
ALIGNERR = 0
INGOODOCTETS = 0
INBADOCTETS = 0
UNDERSIZE = 0
FRAGMENTS = 0
IN64OCTETS = 0
IN127OCTETS = 0
IN255OCTETS = 0
IN511OCTETS = 0
IN1023OCTETS = 0
INMAXOCTETS = 0
JABBER = 0
OVERSIZE = 0
INDISCARDS = 0
INFILTERED = 0

Egress statistics for port 2.
OUTUNICASTS = 0
OUTBROADCASTS = 0
OUTPAUSE = 0
OUTMUILICASTS = 0
OUTFCSERR = 0
OUTOCTETS = 0
OUT64OCTETS = 0
OUT127OCTETS = 0
OUT255OCTETS = 0
OUT511OCTETS = 0
```

```
OUT1023OCTETS = 0
OUTMAXOCTETS = 0
COLLISIONS = 0
LATE = 0
EXECESSIVE = 0
multiple = 0
SINGLE = 0
DEFERRED = 0
OUTFILTERED = 212
OUTDISCARDS = 0

=============================== Port 3 ==================
=============

Statistics for port 3:
Port 3 = 2(TLAN) Back plane
Total good frames received = 0
Total good frames transmitted = 0
vlanStatPrint:  writing data d003 register = 1b address = 1d

Ingress statistics for port 3.
INUNICASTS = 0
INBROADCASTS = 0
INPAUSE = 0
INMUILICASTS = 0
INFCSERR = 0
ALIGNERR = 0
INGOODOCTETS = 0
INBADOCTETS = 0
UNDERSIZE = 0
FRAGMENTS = 0
IN64OCTETS = 0
IN127OCTETS = 0
IN255OCTETS = 0
IN511OCTETS = 0
IN1023OCTETS = 0
INMAXOCTETS = 0
JABBER = 0
OVERSIZE = 0
INDISCARDS = 0
INFILTERED = 0

Egress statistics for port 3.
OUTUNICASTS = 0
OUTBROADCASTS = 0
OUTPAUSE = 0
OUTMUILICASTS = 0
```

```
OUTFCSERR = 0
OUTOCTETS = 0
OUT64OCTETS = 0
OUT127OCTETS = 0
OUT255OCTETS = 0
OUT511OCTETS = 0
OUT1023OCTETS = 0
OUTMAXOCTETS = 0
COLLISIONS = 0
LATE = 0
EXECESSIVE = 0
multiple = 0
SINGLE = 0
DEFERRED = 0
OUTFILTERED = 212
OUTDISCARDS = 0

=============================== Port 4 ==================
=============

Statistics for port 4:
Port 4 = CE Top face plate connector for CPPM only
Total good frames received = 0
Total good frames transmitted = 0
vlanStatPrint:  writing data d004 register = 1b address = 1d

Ingress statistics for port 4.
INUNICASTS = 0
INBROADCASTS = 0
INPAUSE = 0
INMUILICASTS = 0
INFCSERR = 0
ALIGNERR = 0
INGOODOCTETS = 0
INBADOCTETS = 0
UNDERSIZE = 0
FRAGMENTS = 0
IN64OCTETS = 0
IN127OCTETS = 0
IN255OCTETS = 0
IN511OCTETS = 0
IN1023OCTETS = 0
INMAXOCTETS = 0
JABBER = 0
OVERSIZE = 0
INDISCARDS = 0
INFILTERED = 0
```

```
Egress statistics for port 4.
OUTUNICASTS = 0
OUTBROADCASTS = 0
OUTPAUSE = 0
OUTMUILICASTS = 0
OUTFCSERR = 0
OUTOCTETS = 0
OUT64OCTETS = 0
OUT127OCTETS = 0
OUT255OCTETS = 0
OUT511OCTETS = 0
OUT1023OCTETS = 0
OUTMAXOCTETS = 0
COLLISIONS = 0
LATE = 0
EXECESSIVE = 0
multiple = 0
SINGLE = 0
DEFERRED = 0
OUTFILTERED = 1935988
OUTDISCARDS = 0

=============================== Port 5 ==================
=============

Statistics for port 5:
Port 5 = CT Second from top face plate connector for CPPM
only
Total good frames received = 0
Total good frames transmitted = 0
vlanStatPrint:  writing data d005 register = 1b address = 1d

Ingress statistics for port 5.
INUNICASTS = 0
INBROADCASTS = 0
INPAUSE = 0
INMUILICASTS = 0
INFCSERR = 0
ALIGNERR = 0
INGOODOCTETS = 0
INBADOCTETS = 0
UNDERSIZE = 0
FRAGMENTS = 0
IN64OCTETS = 0
IN127OCTETS = 0
IN255OCTETS = 0
```

```
IN511OCTETS = 0
IN1023OCTETS = 0
INMAXOCTETS = 0
JABBER = 0
OVERSIZE = 0
INDISCARDS = 0
INFILTERED = 0

Egress statistics for port 5.
OUTUNICASTS = 0
OUTBROADCASTS = 0
OUTPAUSE = 0
OUTMUILICASTS = 0
OUTFCSERR = 0
OUTOCTETS = 0
OUT64OCTETS = 0
OUT127OCTETS = 0
OUT255OCTETS = 0
OUT511OCTETS = 0
OUT1023OCTETS = 0
OUTMAXOCTETS = 0
COLLISIONS = 0
LATE = 0
EXECESSIVE = 0
multiple = 0
SINGLE = 0
DEFERRED = 0
OUTFILTERED = 72566
OUTDISCARDS = 0

=============================== Port 6 ==================
=============

Statistics for port 6:
Port 6 = 1E Second from bottom face plate connector
Total good frames received = 44008
Total good frames transmitted = 41323
vlanStatPrint:  writing data d006 register = 1b address = 1d

Ingress statistics for port 6.
INUNICASTS = 1319305
INBROADCASTS = 1878703
INPAUSE = 0
INMUILICASTS = 57266
INFCSERR = 0
ALIGNERR = 0
INGOODOCTETS = 301023763
```

```
INBADOCTETS = 0
UNDERSIZE = 0
FRAGMENTS = 0
IN64OCTETS = 1706931
IN127OCTETS = 959311
IN255OCTETS = 509510
IN511OCTETS = 67421
IN1023OCTETS = 11984
INMAXOCTETS = 117
JABBER = 0
OVERSIZE = 0
INDISCARDS = 0
INFILTERED = 110962

Egress statistics for port 6.
OUTUNICASTS = 1220952
OUTBROADCASTS = 19
OUTPAUSE = 0
OUTMUILICASTS = 0
OUTFCSERR = 0
OUTOCTETS = 82880029
OUT64OCTETS = 386212
OUT127OCTETS = 834572
OUT255OCTETS = 163
OUT511OCTETS = 14
OUT1023OCTETS = 9
OUTMAXOCTETS = 1
COLLISIONS = 0
LATE = 0
EXECESSIVE = 0
multiple = 0
SINGLE = 0
DEFERRED = 0
OUTFILTERED = 0
OUTDISCARDS = 0

=============================== Port 7 ==================
=============

Statistics for port 7:
Port 7 = 2T Bottom face plate connector
Total good frames received = 1162
Total good frames transmitted = 5868
vlanStatPrint:  writing data d007 register = 1b address = 1d

Ingress statistics for port 7.
INUNICASTS = 0
```

```
INBROADCASTS = 66698
INPAUSE = 0
INMUILICASTS = 0
INFCSERR = 0
ALIGNERR = 0
INGOODOCTETS = 4268672
INBADOCTETS = 0
UNDERSIZE = 0
FRAGMENTS = 0
IN64OCTETS = 66698
IN127OCTETS = 0
IN255OCTETS = 0
IN511OCTETS = 0
IN1023OCTETS = 0
INMAXOCTETS = 0
JABBER = 0
OVERSIZE = 0
INDISCARDS = 0
INFILTERED = 0

Egress statistics for port 7.
OUTUNICASTS = 23
OUTBROADCASTS = 5841
OUTPAUSE = 0
OUTMUILICASTS = 4
OUTFCSERR = 0
OUTOCTETS = 380300
OUT64OCTETS = 5832
OUT127OCTETS = 6
OUT255OCTETS = 28
OUT511OCTETS = 1
OUT1023OCTETS = 0
OUTMAXOCTETS = 1
COLLISIONS = 0
LATE = 0
EXECESSIVE = 0
multiple = 0
SINGLE = 0
DEFERRED = 0
OUTFILTERED = 0
OUTDISCARDS = 0

=============================== Port 8 ==================
=============

Statistics for port 8:
Port 8 = MSP Internal
```

```
Total good frames received = 6870
Total good frames transmitted = 2558
vlanStatPrint:  writing data d008 register = 1b address = 1d

Ingress statistics for port 8.
INUNICASTS = 1198
INBROADCASTS = 5672
INPAUSE = 0
INMUILICASTS = 0
INFCSERR = 0
ALIGNERR = 0
INGOODOCTETS = 1894704
INBADOCTETS = 0
UNDERSIZE = 0
FRAGMENTS = 0
IN64OCTETS = 5778
IN127OCTETS = 1
IN255OCTETS = 0
IN511OCTETS = 2
IN1023OCTETS = 1
INMAXOCTETS = 1088
JABBER = 0
OVERSIZE = 0
INDISCARDS = 0
INFILTERED = 0

Egress statistics for port 8.
OUTUNICASTS = 1222
OUTBROADCASTS = 66868
OUTPAUSE = 0
OUTMUILICASTS = 4
OUTFCSERR = 0
OUTOCTETS = 4362790
OUT64OCTETS = 68057
OUT127OCTETS = 7
OUT255OCTETS = 28
OUT511OCTETS = 1
OUT1023OCTETS = 0
OUTMAXOCTETS = 1
COLLISIONS = 0
LATE = 0
EXECESSIVE = 0
multiple = 0
SINGLE = 0
DEFERRED = 0
OUTFILTERED = 0
OUTDISCARDS = 0
```

```
=============================== Port 9 ===================
=============

Statistics for port 9:
Port 9 = CSP Internal
Total good frames received = 42205
Total good frames transmitted = 64233
vlanStatPrint:  writing data d009 register = 1b address = 1d

Ingress statistics for port 9.
INUNICASTS = 1221850
INBROADCASTS = 4
INPAUSE = 0
INMUILICASTS = 0
INFCSERR = 0
ALIGNERR = 0
INGOODOCTETS = 82939549
INBADOCTETS = 0
UNDERSIZE = 0
FRAGMENTS = 0
IN64OCTETS = 387027
IN127OCTETS = 834632
IN255OCTETS = 168
IN511OCTETS = 17
IN1023OCTETS = 9
INMAXOCTETS = 1
JABBER = 0
OVERSIZE = 0
INDISCARDS = 0
INFILTERED = 808

Egress statistics for port 9.
OUTUNICASTS = 1208407
OUTBROADCASTS = 1878752
OUTPAUSE = 0
OUTMUILICASTS = 57266
OUTFCSERR = 0
OUTOCTETS = 284868507
OUT64OCTETS = 1661926
OUT127OCTETS = 913202
OUT255OCTETS = 509499
OUT511OCTETS = 51951
OUT1023OCTETS = 7829
OUTMAXOCTETS = 18
COLLISIONS = 0
LATE = 0
```

```
EXECESSIVE = 0
multiple = 0
SINGLE = 0
DEFERRED = 0
OUTFILTERED = 0
OUTDISCARDS = 0
```

## sshKeyShow

Syntax:
**`sshKeyShow`**

Display the SSH key.

The following example shows command output.

```
oam> sshKeyShow
justinsnewhost (47.11.214.83)
Active SSH key fingerprint:  D4:75:B7:03:6E:56:12:61:7D:8
B:D7:78:F3:63:73:E1
```

## swversionshow

Syntax:
**`swversionshow`**

Display the software version.

The following example shows command output.

```
oam> swversionshow
BOOTCODE: MGCBad12
CSP: MGCCAD04
MSP: MGCMAA04
FPGA: MGCFAA08
APPLICATION: MGCAAA02
DB1:  DSP2AA05
DB2:  NONE
```

## testalarm

Syntax:
**`testalarm`**

Send an SNMP trap.

The following example shows command output.

```
oam> testalarm

setting up alarm structure

Calling snmpTrap

SNMP traps are successfully sent

oam> version

VxWorks (for Chagall) version 5.5.1.

Kernel:  WIND version 2.6.

Made on Feb 2 2007, 03:42:05.

Boot line:
ata=0,0(0,0):/p/mainos.sys e=47.11.214.83:ffffff00
g=47.11.214.1 f=0x22 tn=justinsnewhost s=/p/startup1
o=eln
```

## vgwcardshow

Syntax:

**`vgwcardshow cardnum`**

Display all channel information for the specified card.

The following table describes the command parameters.

**Table 100**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| cardnum | – | Card number |

The following example shows command output.

```
oam> vgwcardshow 11
VGW Service is:  Enabled

Chan ChanState DspMode Codec Tn Reg AirTime rxTsap txTsap
RFC 2833 SRTP
---- ---------- --------- ---------- ------ --- -------
-------------------- -------------------- ------------
---------------
32 Idle Closed n/a 0x142c yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
33 Idle Closed n/a 0x142d yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
```

```
n/a Tx n/a Enabled
34 Idle Closed n/a 0x142e yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
35 Idle Closed n/a 0x142f yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
36 Idle Closed n/a 0x146c yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
37 Idle Closed n/a 0x146d yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
38 Idle Closed n/a 0x146e yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
39 Idle Closed n/a 0x146f yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
40 Idle Closed n/a 0x14ac yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
41 Idle Closed n/a 0x14ad yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
42 Idle Closed n/a 0x14ae yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
43 Idle Closed n/a 0x14af yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
44 Idle Closed n/a 0x14ec yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
45 Idle Closed n/a 0x14ed yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
46 Idle Closed n/a 0x14ee yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
47 Idle Closed n/a 0x14ef yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
48 Idle Closed n/a 0x152c yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
49 Idle Closed n/a 0x152d yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
50 Idle Closed n/a 0x152e yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
51 Idle Closed n/a 0x152f yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
52 Idle Closed n/a 0x156c yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
53 Idle Closed n/a 0x156d yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
54 Idle Closed n/a 0x156e yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
55 Idle Closed n/a 0x156f yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
56 Idle Closed n/a 0x15ac yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
```

```
57 Idle Closed n/a 0x15ad yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
58 Idle Closed n/a 0x15ae yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
59 Idle Closed n/a 0x15af yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
60 Idle Closed n/a 0x15ec yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
61 Idle Closed n/a 0x15ed yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
62 Idle Closed n/a 0x15ee yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
63 Idle Closed n/a 0x15ef yes 0 0.0.0.0:0000 0.0.0.0:0000
Rx n/a Tx n/a Enabled
```

## vgwshow

Syntax:
**`vgwshow`**

Show information about busy channels.

The following example shows command output.

```
oam> vgwshow
VGW Service is:  Enabled

Chan ChanState DspMode Codec Tn Reg AirTime rxTsap txTsap
RFC 2833 SRTP
---- --------- -------- --------- ----- --- -------
-------------------- -------------------- -------------
------------
No channel is used.
```

## vgwshowall

Syntax:
**`vgwshowall`**

Display information about all channels.

The following example shows command output.

```
oam> vgwshowall
VGW Service is:  Enabled

Chan ChanState DspMode Codec Tn Reg AirTime rxTsap txTsap
RFC 2833 SRTP
```

```
---- --------- -------- --------- ----- --- -------
------------------- ------------------- -----------
---------------
32 Idle Closed n/a 0x142c yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
33 Idle Closed n/a 0x142d yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
34 Idle Closed n/a 0x142e yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
35 Idle Closed n/a 0x142f yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
36 Idle Closed n/a 0x146c yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
37 Idle Closed n/a 0x146d yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
38 Idle Closed n/a 0x146e yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
39 Idle Closed n/a 0x146f yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
40 Idle Closed n/a 0x14ac yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
41 Idle Closed n/a 0x14ad yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
42 Idle Closed n/a 0x14ae yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
43 Idle Closed n/a 0x14af yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
44 Idle Closed n/a 0x14ec yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
45 Idle Closed n/a 0x14ed yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
46 Idle Closed n/a 0x14ee yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
47 Idle Closed n/a 0x14ef yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
48 Idle Closed n/a 0x152c yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
49 Idle Closed n/a 0x152d yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
50 Idle Closed n/a 0x152e yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
51 Idle Closed n/a 0x152f yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
52 Idle Closed n/a 0x156c yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
53 Idle Closed n/a 0x156d yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
```

```
54 Idle Closed n/a 0x156e yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
55 Idle Closed n/a 0x156f yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
56 Idle Closed n/a 0x15ac yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
57 Idle Closed n/a 0x15ad yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
58 Idle Closed n/a 0x15ae yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
59 Idle Closed n/a 0x15af yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
60 Idle Closed n/a 0x15ec yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
61 Idle Closed n/a 0x15ed yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
62 Idle Closed n/a 0x15ee yes 0 0.0.0.0:0000 0.0.0.0:0000 Rx
n/a Tx n/a Enabled
63 Idle Closed n/a 0x15ef yes 0 0.0.0.0:0000 0.0.0.0:0000
Rx n/a Tx n/a Enabled
```

# Call Server commands

This section lists commands you can enter from the Call Server overlays.

## LD 32

Access the following commands in LD 32.

### ECNT CARD

Syntax:
**`ECNT CARD <card TN> <customer>`**

Print the number of registered and unregistered IP Phones for the specified card. If you specify the customer, the count is specific to that customer. Otherwise, the count is across all customers. If you enter no parameters, the count prints for all zones.

The following table describes the command parameters.

**Table 101**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| card TN | l s c u | Terminal Number for the card.<br><br>Enter a partial TN (for example, l or l s) which then prints the count for that parameter. You cannot specify a customer in this case.<br><br>On the Communication Server 1000 (CS 1000) Option 11C or CSE 1000 systems, enter the card virtual slot number instead of the l s c format. |
| customer | – | Optional. The card TN is mandatory. |

The following example shows command output.

```
.ecnt card 81
<< Card 81 >>
Number of Registered Ethersets:  5
Number of Unregistered Ethersets:  27

.ecnt card

<< Card 61 >>
Number of Registered Ethersets:  7
Number of Unregistered Ethersets:  9

<< Card 62 >>
Number of Registered Ethersets:  0
Number of Unregistered Ethersets:  5

<< Card 64 >>
Number of Registered Ethersets:  0
Number of Unregistered Ethersets:  23
.
```

## ECNT ZONE

Syntax:

**ECNT ZONE <zoneNum> <customer>**

Prints the number of registered and unregistered IP Phones for the specified zone. If you specify the customer number, the count is specific to that customer (a zone must be specified to enter a customer). Otherwise, the count is across all customers. If you enter no parameters, the count prints for all zones.

The following table describes the command parameters.

**Table 102**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| zone | – | Zone |
| customer | 0 to maximum customer number | Customer number |

The following example shows command output.

```
.ecnt zone 0 0
<< Zone 0 Customer 0 >>
Number of Registered Ethersets:  4
Number of Unregistered Ethersets:  17
```

### ENLC/DISC/DISI

Syntax:

```
ENLC <card TN>
DISC <card TN>
DISI <card TN>
```

The **ENLC** and **DISC** commands enable or disable gateway channels across an entire Voice Media Gateway Card (VGMC).

These commands are also useful to troubleshoot card lockup problems. This command is sent to the VGMC processor. You can determine the state of the card processor based on the response.

The **DISI** command disables the card when all channels are idle. Use this command to remove a card from service with minimal impact to users.

> ***Note:*** When the entire card is disabled, the red LED on the faceplate is on. One enabled channel on the card causes the red LED to turn off.

The following table describes the command parameters.

**Table 103**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| card TN | l s c u | Terminal Number for the card.<br><br>On the CS 1000 Option 11C or CSE 1000 systems, enter the card virtual slot number instead of the l s c format. |

The following example shows command output.

```
.enlc 2

XMI001 2 EXUT

XMI002 2 EXUT

.stat 2

00 = UNIT 00 = IDLE (TRK) (TIE )
01 = UNIT 01 = IDLE (TRK) (TIE )
...one line for each channel...
22 = UNIT 22 = IDLE (TRK) (TIE )
23 = UNIT 23 = IDLE (TRK) (TIE )

.disc 2
```

```
.stat 2
00 = UNIT 00 = DSBL (TRK)(TIE )
01 = UNIT 01 = DSBL (TRK)(TIE )
...one line for each channel...
23 = UNIT 23 = DSBL (TRK)(TIE )
```

### ENLU/DISU

Syntax:

**`ENLU <channel TN>`**

**`DISU <channel TN>`**

Enables or disables the specified unit on the VGMC. Use this command to troubleshoot problems with gateway calls. You can force a test call to use a specific set of channels by disabling the others.

The following table describes the command parameters.

**Table 104**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| card TN | l s c u | Terminal Number of the gateway channel for the card |

The following example shows command output.

```
.enlu 2 3

.stat 2 3
IDLE

.disu 2 3

.stat 2 3
DSBL
```

### IDC

Syntax:

**`IDC <card TN>`**

Print information about the VGMC.

This command sends a query to the card XA8051 processor. Use this command to troubleshoot card lockup problems. Based on the response or lack of response, you can determine the state of the card.

The following table describes the command parameters.

**Table 105**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| card TN | l s c | Terminal Number of the card |

The following example shows command output.

```
.idc 2
=> NTVQ55AA R06 NNTMET08DB85 200047
```

## IDU

Syntax:
**`IDU <phone TN>`**

Print the telephone identification information. Use this command to determine the VGMC or Signaling Server to which an IP Phone is currently registered.

You can also use this command to debug card lockup problems. A response indicates end-to-end connectivity between the Call Server and IP Phone and that the VGMC or Signaling Server can pass messages back and forth between them.

The following table describes the command parameters.

**Table 106**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| phone TN | l s c u | Terminal Number of the IP Phone |

The following example shows command output.

```
>ld 32
NPR000
.idu 61 3

I2004 TN 061 0 00 03
TN ID CODE: i2004
NT CODE: NT2K00GI
COLOR CODE: 66
RLS CODE: 0
SER NUM: 760658
SET IP ADR: 010 .001 .001 .006
TPS IP ADR: 047 .011 .214 .052
```

### STAT

Syntax:

**`STAT <card or unit TN>`**

This command provides the status of one or all units on a specified VGMC or of a particular VTN.

The following table describes the command parameters.

**Table 107**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| card or unit TN | l s c<br>l s c u | Terminal Number |

The following example shows output for the VGMC or units.

```
.stat 2 0
IDLE
.stat 2
00 = UNIT 00 = IDLE (TRK)(TIE )
01 = UNIT 01 = IDLE (TRK)(TIE )
...one line for each unit...
21 = UNIT 21 = BUSY
22 = UNIT 22 = UNEQ
23 = UNIT 23 = UNEQ
```

Example output for IP Phone VTNs:

```
.stat 61 1
IDLE REGISTERED
.stat 61 0
UNEQ
.stat 61 3
IDLE UNREGISTERED
```

## LD 80

Access the following commands in LD 80.

### TRAC/TRAK

Syntax:

**`TRAC <TN>`**
**`TRAK <TN>`**

Trace calls associated with the TN. Use this command to determine which gateway channel an IP Phone is connected to or which IP Phone a gateway channel is connected to. Use TRAC on Large Systems and TRAK on Small Systems.

The following table describes the command parameters.

**Table 108**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| TN | l s c u | Terminal Number |

The following example shows command output.

```
.trak 2 0 (track which sets are connected to the gateway
channel with TN 2 0)

ACTIVE TN 002 0 00 00
ORIG 061 0 00 00 0 SCR MARP 0 1111 I2004
TERM 001 0 00 00 0 SCR MARP 0 1234 2616
DIAL DN 1234
MAIN_PM ESTD
TALKSLOT ORIG 8 TERM 10
EES_DATA:
NONE
QUEU NONE
CALL ID 0 81

.trak 61 0 (track the physical TN being used by the i2004
with TN 61 0)

TN 061 0 00 00
KEY 0 SCR MARP ACTIVE TN 061 0 00 00
PHYSICAL TN 002 0 00 00
ORIG 061 0 00 00 0 SCR MARP 0 1111 I2004
TERM 001 0 00 00 0 SCR MARP 0 1234 2616
DIAL DN 1234
MAIN_PM ESTD
TALKSLOT ORIG 8 TERM 10
EES_DATA:
NONE
QUEU NONE
CALL ID 0 81

KEY 1 NUL IDLE
KEY 2 NUL IDLE
KEY 3 NUL IDLE
KEY 4 NUL IDLE
KEY 5 NUL IDLE
KEY 6 NUL IDLE
KEY 7 NUL IDLE
KEY 8 NUL IDLE
KEY 9 NUL IDLE
KEY 10 NUL IDLE
KEY 11 NUL IDLE
KEY 12 NUL IDLE
KEY 13 NUL IDLE
KEY 14 NUL IDLE
KEY 15 NUL IDLE
KEY 16 NUL IDLE
KEY 17 TRN IDLE
KEY 18 AO6 IDLE
KEY 19 CFW IDLE
KEY 20 RGA IDLE
KEY 21 PRK IDLE
KEY 22 RNP IDLE
KEY 23 NUL IDLE
KEY 24 PRS IDLE
KEY 25 CHG IDLE
KEY 26 CPN IDLE
KEY 27 NUL IDLE
KEY 28 NUL IDLE
KEY 29 NUL IDLE
KEY 30 NUL IDLE
KEY 31 NUL IDLE
```

## TRIP

Syntax:
**`TRIP <IP address>`**

Trace an IP address.

The following example shows the output for an IP Phone with IP address 47.166.135.53 in IDLE mode.

```
>ld 80
TRA000
.trip 47.166.135.53

VTN 096 0 03 07
KEY 0 SCR MARP IDLE
KEY 1 NUL IDLE
KEY 2 MSB IDLE
KEY 3 NUL IDLE
KEY 4 NUL IDLE
KEY 5 HNDO IDLE
KEY 6 NUL IDLE
KEY 7 NUL IDLE
KEY 8 NUL IDLE
KEY 9 NUL IDLE
KEY 10 NUL IDLE
KEY 11 ADL IDLE
KEY 12 NUL IDLE
KEY 13 NUL IDLE
KEY 14 NUL IDLE
KEY 15 NUL IDLE
KEY 16 MWK IDLE
KEY 17 TRN IDLE
KEY 18 AO6 IDLE
KEY 19 CFW IDLE
KEY 20 RGA IDLE
KEY 21 PRK IDLE
KEY 22 RNP IDLE
KEY 23 NUL IDLE
KEY 24 PRS IDLE
KEY 25 CHG IDLE
KEY 26 CPN IDLE
KEY 27 NUL IDLE
KEY 28 ADL IDLE
KEY 29 NUL IDLE
KEY 30 NUL IDLE
KEY 31 NUL IDLE
KEY 32 ADL IDLE
KEY 33 ADL IDLE
KEY 34 ADL IDLE
KEY 35 ADL IDLE
KEY 36 ADL IDLE
KEY 37 ADL IDLE
KEY 38 ADL IDLE
KEY 39 ADL IDLE
KEY 40 ADL IDLE
KEY 41 ADL IDLE
KEY 42 ADL IDLE
KEY 43 ADL IDLE
KEY 44 ADL IDLE
KEY 45 ADL IDLE
KEY 46 ADL IDLE
KEY 47 ADL IDLE
KEY 48 ADL IDLE
KEY 49 ADL IDLE
KEY 50 ADL IDLE
KEY 51 ADL IDLE
KEY 52 ADL IDLE
KEY 53 ADL IDLE
KEY 54 ADL IDLE
KEY 55 ADL IDLE
```

The following example shows the output for an IP Phone with IP address 47.166.135.53 in off-hook with dial tone.

```
.trip 47.166.135.53

VTN 096 0 03 07
KEY 0 SCR MARP ACTIVE VTN 096 0 03 07

ORIG VTN 096 0 03 07 KEY 0 SCR MARP CUST 0 DN 3614 TYPE 1230
MEDIA ENDPOINT IP: 47.166.135.53 PORT: 5200
TERM NONE
TDTN 44 SLOT 0 1 PTY SLOT 0 1
DIAL DN NONE
MAIN_PM READY
TALKSLOT NONE
EES_DATA:
NONE
QUEU DIAL
CALL ID 570 19942

KEY 1 NUL IDLE
KEY 2 MSB IDLE
KEY 3 NUL IDLE
KEY 4 NUL IDLE
KEY 5 HNDO IDLE
KEY 6 NUL IDLE
KEY 7 NUL IDLE
KEY 8 NUL IDLE
KEY 9 NUL IDLE
KEY 10 NUL IDLE
KEY 11 ADL IDLE
KEY 12 NUL IDLE
KEY 13 NUL IDLE
KEY 14 NUL IDLE
KEY 15 NUL IDLE
KEY 16 MWK IDLE
KEY 17 TRN IDLE
KEY 18 AO6 IDLE
KEY 19 CFW IDLE
KEY 20 RGA IDLE
KEY 21 PRK IDLE
KEY 22 RNP IDLE
KEY 23 NUL IDLE
KEY 24 PRS IDLE
KEY 25 CHG IDLE
KEY 26 CPN IDLE
KEY 27 NUL IDLE
KEY 28 ADL IDLE
KEY 29 NUL IDLE
KEY 30 NUL IDLE
KEY 31 NUL IDLE
KEY 32 ADL IDLE
KEY 33 ADL IDLE
KEY 34 ADL IDLE
KEY 35 ADL IDLE
KEY 36 ADL IDLE
KEY 37 ADL IDLE
KEY 38 ADL IDLE
KEY 39 ADL IDLE
KEY 40 ADL IDLE
KEY 41 ADL IDLE
KEY 42 ADL IDLE
KEY 43 ADL IDLE
KEY 44 ADL IDLE
KEY 45 ADL IDLE
KEY 46 ADL IDLE
KEY 47 ADL IDLE
KEY 48 ADL IDLE
KEY 49 ADL IDLE
KEY 50 ADL IDLE
KEY 51 ADL IDLE
KEY 52 ADL IDLE
KEY 53 ADL IDLE
KEY 54 ADL IDLE
KEY 55 ADL IDLE
```

The following example shows the output for an IP Phone with IP address 47.166.135.53 with call established.

```
.trip 47.166.135.53

VTN 096 0 03 07
KEY 0 SCR MARP ACTIVE VTN 096 0 03 07

ORIG VTN 096 0 03 07 KEY 0 SCR MARP CUST 0 DN 3614 TYPE 1230
MEDIA ENDPOINT IP: 47.166.135.53 PORT: 5200
TERM VTN 096 0 01 25 KEY 0 SCR MARP CUST 0 DN 3616 TYPE 1230
MEDIA ENDPOINT IP: 47.166.129.191 PORT: 5200
MEDIA PROFILE: CODEC G.711 A-LAW PAYLOAD 20 ms VAD OFF
DIAL DN 3616
MAIN_PM ESTD
TALKSLOT ORIG 10 TERM 42
EES_DATA:
NONE
QUEU NONE
CALL ID 570 19941

KEY 1 NUL IDLE
KEY 2 MSB IDLE
KEY 3 NUL IDLE
KEY 4 NUL IDLE
KEY 5 HNDO IDLE
KEY 6 NUL IDLE
KEY 7 NUL IDLE
KEY 8 NUL IDLE
KEY 9 NUL IDLE
KEY 10 NUL IDLE
KEY 11 ADL IDLE
KEY 12 NUL IDLE
KEY 13 NUL IDLE
KEY 14 NUL IDLE
KEY 15 NUL IDLE
KEY 16 MWK IDLE
KEY 17 TRN IDLE
KEY 18 AO6 IDLE
KEY 19 CFW IDLE
KEY 20 RGA IDLE
KEY 21 PRK IDLE
KEY 22 RNP IDLE
KEY 23 NUL IDLE
KEY 24 PRS IDLE
KEY 25 CHG IDLE
KEY 26 CPN IDLE
KEY 27 NUL IDLE
KEY 28 ADL IDLE
KEY 29 NUL IDLE
KEY 30 NUL IDLE
KEY 31 NUL IDLE
KEY 32 ADL IDLE
KEY 33 ADL IDLE
KEY 34 ADL IDLE
KEY 35 ADL IDLE
KEY 36 ADL IDLE
KEY 37 ADL IDLE
KEY 38 ADL IDLE
KEY 39 ADL IDLE
KEY 40 ADL IDLE
KEY 41 ADL IDLE
KEY 42 ADL IDLE
KEY 43 ADL IDLE
KEY 44 ADL IDLE
KEY 45 ADL IDLE
KEY 46 ADL IDLE
KEY 47 ADL IDLE
KEY 48 ADL IDLE
KEY 49 ADL IDLE
KEY 50 ADL IDLE
KEY 51 ADL IDLE
KEY 52 ADL IDLE
KEY 53 ADL IDLE
KEY 54 ADL IDLE
KEY 55 ADL IDLE
```

## LD 117

Access the following commands in LD 117.

### STIP

The STIP command has six variations that print IP Phone–related IP information that requires modification:

- **`STIP TN <TN>`**
  Display the Resource Locator Module information for the specified TN or group of TNs.
- **`STIP TYPE <type>`**
  Display the Resource Locator Module information for the specified TN type.
- **`STIP ZONE <zone>`**
  Display the Resource Locator Module information for the specified zone.
- **`STIP NODE <node>`**
  Display the Resource Locator Module information for the specified node.
- **`STIP HOSTIP <host IP address>`**
  Display the Resource Locator Module information corresponding to the specified host IP.
- **`STIP TERMIP <IP address>`**
  Display the Resource Locator Module information corresponding to the specified IP Phone public IP address for signaling.

In the printed output for this command, the public IP address and port for the signaling appear in the SIGNALING IP column. For NAT telephones, the private IP address for the signaling appears below it in parentheses.

The following example shows command output.

```
=> stip tn 61 0

TN type HWID STATUS HOSTIP SIGNALING IP CODEC BDWITH
61 0 0 0 i2004 MAC: REG 47.11.216.49 30.1.1.10 - 0
1800603876c79d6600
G711u noVAD 1904

G711a noVAD 1904

G729AB 470

=> stip tn 61 1

TN type HWID STATUS HOSTIP SIGNALING IP CODEC BDWITH
61 0 0 1 i2004 MAC: REG 47.11.216.50 30.1.1.100:1250 - 0
18006038763 8e06600 (192.168.1.13)
G711u noVAD 1904

G711u noVAD 1904

=> stip termip 30.1.1.100

TN type HWID STATUS HOSTIP Signaling IP CODEC BDWITH
61 0 0 1 i2004 MAC: REG 47.11.216.50 30.1.1.100:1250 - 0
180060387638e06600 (192.168.1.13)
G711u noVAD 1904

G711a noVAD 1904
61 0 0 2 i2004 MAC: REG 47.11.216.50 30.1.1.100:1248 - 0
180060387638ee6600 (192.168.1.12) G711u noVAD 1904

G711a noVAD 1904
```

## PRT

The PRT command has two variations that print IP Phone-related IP information that requires modification:

- **`PRT IPDN <IP address>`**
  Print DNs associated with an IP Phone public IP address for signaling.
- **`PRT DNIP`**
  Print IP addresses associated with a DN.

The following example shows the output for the PRT IPDN command (the public IP address and port are printed followed by the private IP address and port in parentheses).

```
=> prt ipdn 30.1.1.100

Signaling IP 30.1.1.100:1248 (192.168.1.12)
Media IP 30.1.1.100:1246 (192.168.1.12:5200)
CUST 00 TN 061-02 TYPE i2002 ZONE 000 REG
Key DN CPND Name
---------------------------------------------
00 SCR 2013 I2002_Cust_0
01 SCR 2001 I2002_Cust_0

Signaling IP 30.1.1.100:1250 (192.168.1.13)
Media IP 30.1.1.100:1252 (192.168.1.12:5200)
CUST 00 TN 061-01 TYPE i2004 ZONE 000 REG
Key DN CPND Name
---------------------------------------------
01 SCR 2041
05 SCR 2042

=> prt ipdn 30.1.1.10

Signaling IP 30.1.1.10:5000
Media IP 30.1.1.10:5200
CUST 00 TN 061-00 TYPE i2004 ZONE 000 REG
Key DN CPND Name
---------------------------------------------
01 SCR 8001
```

The following example shows the output for the PRT DNIP command (in the IP Address column, the public IP address and port are printed followed by the private IP address and port in parentheses).

```
=> prt dnip 8001

CUST 00 DN 8001
TN Type Key Signaling IP Address Media IP Address Zone
Status
-------------------------------------------------------
-------------------------------------
061-00 i2004 01 SCR 30.1.1.10:5000 30.1.1.10:5200 000 REG

=> prt dnip 2041

CUST 00 DN 2041
```

```
TN Type Key Signaling IP Address Media IP Address Zone
Status
-------------------------------------------------------
--------------------------------------
061-01 i2004 01 SCR 30.1.1.100:1250 30.1.1.100:1252 000 REG
(192.168.1.13) (192.168.1.13:5200)
```

## STAT IP

The STAT IP command has six variations that print IP-Phone related IP information that requires modification:

- **STAT IP TN <TN>**
  Print the Resource Locator Module information for the specified TN or group of TNs.
- **STAT IP TYPE <type>**
  Print the Resource Locator Module information for the specified TN type.
- **STAT IP ZONE <zone>**
  Print the Resource Locator Module information for the specified zone.
- **STAT IP NODE <node>**
  Print the Resource Locator Module information for the specified node.
- **STAT IP HOSTIP <host IP address>**
  Print the Resource Locator Module information that corresponds to the specified host IP address.
- **STAT IP TERMIP <IP address>**
  Print the Resource Locator Module information that corresponds to the specified IP Phone public IP address for signaling.

In the printed output for this command, the public IP address and port for the signaling appear in the SIGNALING IP column. For NAT telephones, the private IP address for the signaling appears below it in parentheses.

The following example shows command output.

```
=> stat ip type i2004

TN type HWID STATUS HOSTIP SIGNALING IP CODEC BDWITH
61 0 0 0 i2004 MAC: REG 47.11.216.49 30.1.1.10:5000 - 0
1800603876c79d6600 G711u noVAD 1904
G711a noVAD 1904
G729AB 470
61 0 0 1 i2004 MAC: REG 47.11.216.50 30.1.1.100:1250 - 0
18006038763 8e06600 (192.168.1.13) G711u noVAD 1904
G711a noVAD 1904
```

## PDT commands

You can run the PDT commands at the PDT shell on the Call Server. You may need to load the symbol table (pdt> symload) to gain access to these commands.

### rudpShow

Syntax:

**`rudpShow`**

Display information about the RUDP connections currently active on the Call Server.

The following example shows command output.

```
pdt> rudpShow

+----------+---------+-------+----+-----+----------+-
---------+
| Port ID | Src IP |Src Port| FD | Task| Data 1 | Data 2 |
+----------+---------+-------+----+-----+----------+-
---------+
|537400104 |0xc0a8010a|15000 |28 |13312|0 |0 |
+----------+---------+-------+----+-----+----------+-
---------+
+----------+---------+-------+-------------+-------
---+----------+--------+
|Connect ID| Dst IP |Dst Port| Status | Msg rcv | Msg sent |
Retries|
+----------+---------+-------+-------------+-------
---+----------+--------+
|537516520 |0xc0a8010e|15000 |ESTABLISHED <->|20596
|1057753 |0 |
+----------+---------+-------+-------------+-------
---+----------+--------+

value = 0 = 0x0
```

### rlmShow

Syntax:

**`rlmShow TN`**

Display information from the Resource Locator Module (RLM) about an administered VGMC gateway channel and Internet terminal device.

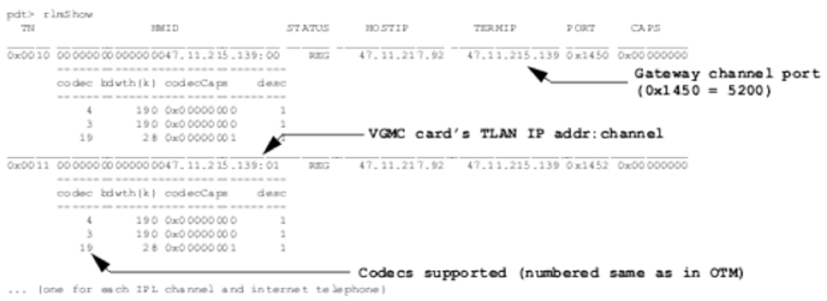
Use this command to check that all gateway channels and IP Phones associated with a VGMC node display the same list of codecs. If the same codec lists do not appear, it is likely that the Card Properties was not downloaded from OTM/EM to all cards in that node. For more information about this issue, see "Cannot make calls with particular zone and codec configurations" (page 38).

The following table describes the command parameters.

**Table 109**
**Command parameters**

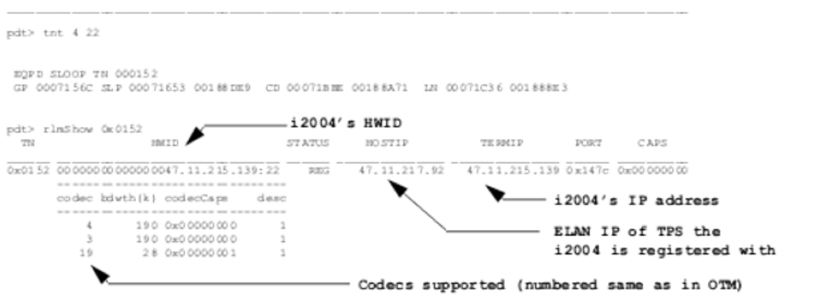
| Parameter | Value | Description |
|---|---|---|
| TN | NxNNNN | Optional. Terminal Number for the device in hexadecimal format.<br><br>If you do not enter this parameter, the data for all devices prints. |

The following example shows command output.



```
pdt> rlmShow
  TN                HWID                 STATUS    HOSTIP        TERMIP         PORT     CAPS
-------------------------------------------------------------------------------------------------
0x0010 000000000000000047.11.215.139:00   REG    47.11.217.92  47.11.215.139  0x1450  0x00000000
       ---------------------------------
       codec bdwth(k) codecCaps     desc
       ---------------------------------
           4      190 0x00000000       1
           3      190 0x00000000       1
          19       28 0x00000001       1
-------------------------------------------------------------------------------------------------
0x0011 000000000000000047.11.215.139:01   REG    47.11.217.92  47.11.215.139  0x1452  0x00000000
       ---------------------------------
       codec bdwth(k) codecCaps     desc
       ---------------------------------
           4      190 0x00000000       1
           3      190 0x00000000       1
          19       28 0x00000001       1
... (one for each IPL channel and internet telephone)
```

Gateway channel port
(0x1450 = 5200)
VGMC card's TLAN IP addr:channel
Codecs supported (numbered same as in OTM)



```
-------------------------------------------------------------------------------------------------
pdt> tnt 4 22

 EQPD SLOOP TN 000152
 GP 0007156C SLP 00071653 00188DE9  CD 00071BBE 00188A71  LN 00071C36 00188BE3

pdt> rlmShow 0x0152
  TN                HWID                 STATUS    HOSTIP        TERMIP          PORT     CAPS
-------------------------------------------------------------------------------------------------
0x0152 000000000000000047.11.215.139:22   REG    47.11.217.92  47.11.215.139  0x147c  0x00000000
       ---------------------------------
       codec bdwth(k) codecCaps     desc
       ---------------------------------
           4      190 0x00000000       1
           3      190 0x00000000       1
          19       28 0x00000001       1
```

i2004's HWID
i2004's IP address
ELAN IP of TPS the
i2004 is registered with
Codecs supported (numbered same as in OTM)

## uZoneShow

Syntax:

`uZoneShow zone`

Display the amount of intra- (IN) and inter- (OUT) zone bandwidth used for the specified zone. Print the same bandwidth usage data as that printed by LD 117.

The following example shows command output.

```
pdt> uZoneShow

Zone 0 BW used:  IN = 254(LO) OUT = 0(LO)
value = 43 = 0x2B
pdt> uZoneShow 1

Zone 1 BW used:  IN = 0(LO) OUT = 0(LO)
value = 41 = 0x29
```

## Attendant Monitoring Tool in PDT (CS 1000 Release 5.5 only)

Attendant monitoring commands in PDT were introduced in CS 1000 Release 5.5 to monitor the attendant status when it changes service or state. This is useful to address a critical problem observed with the attendant feature, whereby the attendant enters NITE service unexpectedly.

### attnMonOn

This command turns on the attendant monitor and displays the following information whenever the attendant changes state:

- customer number
- attendant number that recently changed in service (IN or OUT SRVC)
- number of attendants in service
- customer service (DAY or NITE SRVC)
- attendant number and key pressed (NITE key or Position Busy key) on the attendant causing the customer to move into DAY or NITE service.

The following example shows command output.

```
pdt> attnMonOn

ATTN Monitor is on

value = 0 = 0x0

Any change in the status of attendant

DIAG200

DIAG0200:  CUST#<CUSTNO> ATTN#<ATTNUM>OUT SRVC <No.  of
ATTNs in service> LEFT

DIAG201
```

```
DIAG0201: CUST# <CUSTNO> NITE (or DAY) SRVC, ATTN#<
ATTNUM> NITE (or PBSY) KEY PRESSED
```

### attnMonOff

This command turns off the attendant monitor.

The following example shows command output.

```
pdt> attnMonOff

ATTN Monitor is off

value = 0 = 0x0
```

### Call Register Monitoring Tool in PDT (sl1crShow)

Call Register Monitoring was introduced in CS 1000 Release 5.5 to address Call Register leakage by providing the status of all Call Registers at a particular time. The status is based on the MAINPM and AUXPM values of the Call Register.

The **`sl1crShow`** command has the following variations that provide Call Register Monitoring:

- **`sl1crShow <interval>`**
  Print the number of Call Registers for various nonzero MAINPM values, repeating at the specified interval (in seconds).
- **`sl1crShow <mainpm> <auxpm> <interval>`**
  Print the number of Call Registers with the specified mainpm and auxpm, and repeated at the specified interval (in seconds).
- **`sl1crShowStop`**
  Deactivate commands 2, 3, and 4.
- **`sl1crShowHelp`**
  Print help for the Call Register Monitoring commands.

The following table describes the parameters input with the sl1crShow command.

**Table 110**
**Command parameters**

| Parameter | Value | Description |
|---|---|---|
| interval | 1–3600 | Time interval (seconds) between successive printing |

**Table 110**
**Command parameters (cont'd.)**

| Parameter | Value | Description |
|---|---|---|
| mainpm | 0–27 | MAINPM value for corresponding AUXPM printing |
| auxpm | 0 to maximum AUXPM for this MAINPM | AUXPM value for which the number of Call Registers are printed |

The following example shows the output for the various sl1crShow commands where a telephone calls another telephone and the called telephone answers.

```
pdt> sl1crShow
MAINPM NCR
IDLE 791
ESTABLISHED 2
SPECIAL 5
Total CR Count = 798
value = 1 = 0x1

pdt> sl1crShow 10
MAINPM NCR
IDLE 791
ESTABLISHED 2
SPECIAL 5
Total CR Count = 798
value = 0 = 0x0
The status will keep printing after every 10 seconds.

pdt>sl1crShow 12 5
MAINPM = SPECIAL
AUXPM NCR
AML_MAINT 1
PRA_MSG_CR 2
PRA_ROSE_CR 2
Total AUXPM CR Count = 5
value = 0 = 0x0
The status will keep printing after every 5 seconds.

pdt>sl1crShow 12 37 15
MAINPM = SPECIAL
AUXPM = AML_MAINT
The Call Register count = 1
value = 0 = 0x0
The status will keep printing after every 15 seconds.

pdt> sl1crShowStop
```

```
value = 0 = 0x0
pdt> sl1crShowHelp
sl1crShow :
Usage is to be one among the following :
sl1crShow [interval]
sl1crShow [mainpm] [interval]
sl1crShow [mainpm] [auxpm] [interval]

Prints the number of Call Registers for the specified mainpm
and auxpm values at the specified intervals

INPUT: interval - Time interval between successive
printings
[range - 1 to 3600 (seconds)]
mainpm - MAINPM value for corresponding AUXPM printing
[range - 0 to 27]
auxpm - AUXPM value for which the number of Call Registers
will be printed [range - 0 to MAX AUXPM of that MAINPM]
sl1crShowStop - to stop printing the Call Register status.
value = 0 = 0x0
```

## VNS BUG ERR debug tool in PDT (CS 1000 Release 5.5 only)

The VNS tool introduced in CS 1000 Release 5.5 provides extensive diagnostic data to be printed with the VNS ERROR and VNS BUG. You can use this tool to troubleshoot various call scenarios involving VNS. With the vnsdebug command on, detailed information regarding the VNS BUG/ERR prints on the TTY. This command also ensures that a RAS prints with the VNS ERROR.

The **`vnsdebug`** command has the following variations that provide VNS debugging:

- **`vnsdebug`**
  Print the current status of the vnsdebug command.
- **`vnsdebug ?`** or **`vnsdebug help`** or **`vnsdebug <any junk value>`**
  Print help on how to use the vnsdebug command.
- **`vnsdebug off`**
  Turn off enhanced VNS BUG/ERR printing.
- **`vnsdebug on`**
  Turn on enhanced VNS BUG/ERR printing on the TTY.

  ***Note:*** For an invalid parameter entered after the vnsdebug command (such as in the following example in the line `pdt> vnsdebug qwert`), the vnsdebug command output prints on the TTY.

The following example shows command output.

```
pdt> symload
Loading symbols from "c:/p/sl1/res.sym"
pdt> vnsdebug
vnsdebug enhanced print currently:  OFF
value = 0 = 0x0
pdt> vnsdebug ?
Usage:  vnsdebug [on|off]
value = 0 = 0x0
pdt> vnsdebug help
Usage:  vnsdebug [on|off]
value = 0 = 0x0
pdt> vnsdebug qwert
Usage:  vnsdebug [on|off]
value = -1 = 0xFFFFFFFF
```

The following example shows the output of VNS BUG/ERR when the vnsdebug command is off.

```
pdt> vnsdebug off
vnsdebug enhanced print:  OFF
value = 0 = 0x0
BUG5182
BUG5182 :  TASK= 46 TN= 00001548 CK= 0000A0D0 19:27:14
23/05/2007 BUG5182 + 107FC964 20D99288 20D91272 20D907CE
20D95D02
BUG5182 + 20D903E0 10BB0C62 10BA22EA 10B9278C 10B8F6E2
BUG5182 + 10B8F602 10B8F5B0 10B8524E 109215E0 10920232
BUG5182 + 109200C2 108FB442 108FAC24 107D342E 107C7DFE
BUG5182 + 107C6218 107C37B4 107C33A2 107C26EA 107BEFCA
BUG5182 + 10F3E804 104113DE 104105A0 1040FE02 1040F456
BUG5182 + 106EFDA0 106EF6E4 106E96EA 106E4D36 110C66C4
BUG5182 + 110C61F4 110C610A 110C5D80 110C0D52 110BA26E
BUG5182 1=00164CB3 00000607 00008218 00001548 00000000 0
00000000 0000000 0000
0000 0 2000 00000084 0000AAA2 00000000 00000000 00000000
00000000 00000000
BUG5182 2=00000259 00000000
BUG5182 5=000001A1 00000002 00008300 00000000 00000000
00000000 00000000 00164C
B3 00165823 001646FB 0016ACC6 0016ACBB
ERR5427 TASK= 46 TN= 00001548 CK= 0A65F5D3 19:27:14
23/05/2007
ERR5427 1=00164CB3 00000607 00008218 00001548 00000000 0
00000000 00000000 0000
0000 0 2000 00000084 0000AAA2 00000000 00000000 00000000
00000000 00000000
```

```
ERR5427 2=00000259 00000000
ERR5427 5=000001A1 00000002 00008300 00000000 00000000
00000000 00000000 00164CB3 00165823 001646FB 0016ACC6
0016ACBB
```

The following example shows the output of VNS BUG/ERR when the vnsdebug command is on.

```
pdt> vnsdebug on
vnsdebug enhanced print:  ON
value = 0 = 0x0

BUG5182
BUG5182 :  TASK= 46 TN= 00001548 CK= 0000A02F 19:27:56
23/05/2007
BUG5182 + 107FC964 20D99288 20D91272 20D907CE 20D95D02
BUG5182 + 20D903E0 10BB0C62 10BA22EA 10B9278C 10B8F6E2
BUG5182 + 10B8F602 10B8F5B0 10B8524E 109215E0 10920232
BUG5182 + 109200C2 108FB442 108FAC24 107D342E 107C7DFE
BUG5182 + 107C6218 107C37B4 107C33A2 107C26EA 107BEFCA
BUG5182 + 10F3E804 104113DE 104105A0 1040FE02 1040F456
BUG5182 + 106EFDA0 106EF6E4 106E96EA 106E4D36 110C66C4
BUG5182 + 110C61F4 110C610A 110C5D80 110C0D52 110BA26E

BUG5182 CRPTR:
00167E43 00000607 00008218 00001548 00000000 0 00000000
00000000 00000000 0
2000 00000084 0000AAA2 00000000 00000000 00000000 00000000
00000000

BUG5182 VNS_ORIG_INDEX:CRPTR=00000259
VNS_TER_INDEX:CRPTR=00000000
BUG5182 VDNBLOCK:
000001A1 00000002 000082FF 00000000 00000000 00000000
00000000 00167E43 00166393 001671DF 0016ACC6 0016ACBB

ERR5427
107FCBD2 20D991F6 20D9127E 20D907CE 20D95D02
ERR5427 + 20D903E0 10BB0C62 10BA22EA 10B9278C 10B8F6E2
ERR5427 + 10B8F602 10B8F5B0 10B8524E 109215E0 10920232
ERR5427 + 109200C2 108FB442 108FAC24 107D342E 107C7DFE
ERR5427 + 107C6218 107C37B4 107C33A2 107C26EA 107BEFCA
ERR5427 + 10F3E804 104113DE 104105A0 1040FE02 1040F456
ERR5427 + 106EFDA0 106EF6E4 106E96EA 106E4D36 110C66C4
ERR5427 + 110C61F4 110C610A 110C5D80 110C0D52 110BA26E
ERR5427 :TASK= 46 TN= 00001548 CK= 0A674738 19:27:56
23/05/2007
```

```
ERR5427 CRPTR:
00167E43 00000607 00008218 00001548 00000000 0 00000000
00000000 00000000 0 200
0 00000084 0000AAA2 00000000 00000000 00000000 00000000
00000000

ERR5427 VNS_ORIG_INDEX:CRPTR=00000259
VNS_TER_INDEX:CRPTR=00000000
ERR5427 VDNBLOCK:
000001A1 00000002 000082FF 00000000 00000000 00000000
00000000 00167E43 0016639
3 001671DF 0016ACC6 0016ACBB
```

## umcUtility command in PDT (CS 1000 Release 5.5 only)

Syntax:

**`umcUtility <code> [<data>] [<customer>]`**

This feature is a support utility introduced in CS 1000 Release 5.5 to debug and test the Mobile X User Database. Run the command from the PDT shell. Depending on the command parameters, this command can perform various actions. In a multi-customer configuration, you must specify the customer number to work with the correct hash table. This parameter is 0 by default.

Example 1:

```
pdt> umcUtility 0
Usage:  umcUtility <code> [<data>] [<customer>]
customer = 0 by default
<code> <data> <function>

0 Print this help menu
1 0 Print profile summary
1 1 Dump slot info and profile summary
2 0 Turn off debug print
2 1 Turn on debug print
3 slot Print details of a slot
4 Print non-empty slots
(warning:  this may take some time depending on the size of
the data)
5 “DN” Search for a mobile DN
DN must be specified in quotes & must be same as programmed
in UXID prompt in LD11 / LD20.
6 TN Search for a specific TN
TN must be specified in compact form with 0x prefix, eg, for
TN 200 0 2 0, enter 0xC808 value = 45 = 0x2D
value = 45 = 0x2D
```

Example 2:

```
User database profile statistics:
Number of users = 15
Number of slots used = 15
Max count for a slot = 1
Slot with the max count = 510
Slot Count Distribution:
 [7904] slot(s) have [0] users [15] slot(shave [1] users
Database info:
User Database Headpointer = [0x1055b8e8]
Current User Count = [15]
Debug flag @ [0x218be90] or SL1 [0x77a4] = [0x0]
value = 49 = 0x31
```

Example 3:

```
pdt> umcUtility 1 1
Table usage dump:
slot [510] count [1]
slot [525] count [1]
slot [526] count [1]
slot [1071] count [1]
slot [3332] count [1]
slot [3784] count [1]
slot [3972] count [1]
slot [3978] count [1]
slot [4842] count [1]
slot [6013] count [1]
slot [6016] count [1]
slot [6521] count [1]
slot [7279] count [1]
slot [7603] count [1]
slot [7608] count [1]

User database profile statistics:
Number of users = 15
Number of slots used = 15
Max count for a slot = 1
Slot with the max count = 510
Slot Count Distribution:
 [7904] slot(s) have [0] users
 [15] slot(s) have [1] users
Database info:
User Database Headpointer = [0x1055b8e8]
```

```
Current User Count = [15]
Debug flag @ [0x218be90] or SL1 [0x77a4] = [0x0]
value = 49 = 0x31
```

Example 4:

```
pdt> umcUtility 2 1
Debug print has been turned on
value = 32 = 0x20

pdt> umcUtility 5 "868501135"
Searching for MDN [868501135]
search MDN [5868 311a 0005 0000 ]
For this mobile DN, the data is stored in slot [525]
Details:  Slot number [525] UEXT TN [6c01]
MDN [5868 311a 0005 0000 ]
DDN [4163 0000 ]
value = 2 = 0x2
```

# IP Phones

## Special Key Sequences

Use special key sequences (SKS) to retrieve information about IP Phones. You can also reset the IP Phone without unplugging the power supply.

After most of these commands, the display remains in a suspended condition. To return the display to normal, press the Services key twice.

### Manual Configuration Display

The Telephone Options menu on the Services key has a command that displays the Set Info. The Set Info displays the IP Phone IP address, hardware ID, firmware version, TN, node IP address, and node ID (if the node level-password is enabled, the IP address and TN are not displayed).

To see all information that is manually configurable at the IP Phone, enter the following SKS: Mute, Right Arrow, Up Arrow, Left Arrow, Down Arrow, Up Arrow, Mute, Mute.

Press the left softkey (OK) to advance through the configured entries until the display becomes blank. This displays the following information about the IP Phone: DHCP setting, IP address and netmask, default gateway IP address, S1 and S2 IP address, port action and retry count, and VLAN configuration data. When the display becomes blank, press the Services key twice to restore the normal display. If you perform this procedure during an active call, the call and feature states are not affected, but the Transmit (TX) path becomes mute until the display finishes. After the screen becomes blank, press the Mute key again to unmute the Transmit (TX) path.

### Downloadable firmware version display

At any time during normal operation, you can view the current downloadable firmware version. Press Mute, Up Arrow, Down Arrow, Up Arrow, Down Arrow, Up Arrow, Mute, *, 0.

This SKS displays the firmware version, for example, "0602B39". Press any key to exit this mode.

## MAC Display

When the IP Phone powers up, it displays a Nortel Networks prompt for approximately four seconds. After this prompt appears, you have approximately 1.5 seconds to press the first three softkeys, which are directly below the LCD, in order from left to right (1,2,3), followed by the Up Arrow navigation key. After you enter this key sequence, the IP Phone enters the new MAC address viewing session with the "Mac Address" prompt in the context field of the LCD and the prompt "MAC: XXXXXXXXXXXX" on line 1 of the LCD (where XXXXXXXXXXXX is the MAC address of this E2). Additionally, the IP Phone displays 1 softkey function:

Softkey 1 - OK (continue).

Alternatively, when the IP Phone is operational, press the Services key, select Set Info and advance to HWID. The MAC address is the last 12 digits of the displayed number.

## IP Phone reboot

At any time during normal operation, you can reset the IP Phone by pressing Mute, Up Arrow, Down Arrow, Up Arrow, Down Arrow, Up Arrow, Mute, 9, Rls (telephone handset with arrow pointing downward key).

After the telephone resets, it displays the Nortel Networks prompt for four seconds. The telephone attempts to register at a random interval from 0 to 30 seconds after the Nortel Networks prompt disappears. The Connect Svc 1 configuration screen with the TN and node ID appears after the IP Phone connects successfully with the Voice Media Gateway Card (VGMC) connect server.

## RUDP status display and toggling state

By default, the IP Phone RUDP is on. To turn RUDP off for debugging, wait for the "Nortel Networks" prompt at power up; then, press the first three softkeys, which are directly below the LCD, in order from left to right (1,2,3), followed by the green, lower right soft-label key. Each time you enter this sequence, the RUDP state (on or off) switches in the nonvolatile storage on the IP Phone.

Typically, you enter this command only to determine if the IP Phone inadvertently switched to RUDP off, thus preventing it from registering properly with the TPS.

At any time during normal operation, you can view and switch the RUDP state by pressing Mute, Up Arrow, Down Arrow, Up Arrow, Down Arrow, Up Arrow, Mute, *, 2.

The current RUDP state appears, and one softkey is available to switch the state and another to exit.

## UNIStim 3.0

Use the following troubleshooting suggestions for UNIStim 3.0.

### Enhanced diagnostics

With the IP Phone enhanced diagnostics feature, you can locate, examine, and eliminate problems in the UNIStim IP Phone systems. This feature can be used both locally and remotely.

This feature provides two functions. First it detects and logs errors, and can recover the telephone. Second it provides a secure shell (SSH) access to the telephone for administration and debugging.

The error collecting, logging, and recovery function provides task monitor, CPU usage monitor, task stack monitor, memory monitor, and watch dog to monitor and log errors at run time. This feature can also perform a hard reset of the telephone.

This feature provides two levels of diagnostics shell privilege. The first level is the PDT level which is a general user level and provides the full set of PDT commands. The second level is the VxShell level, which is a superuser level, provides the VxWorks native shell (tShell) commands and a full set of PDT commands. To access the PDT level, you require the SSH user ID and password. To access the vxshell level, you require permission on the Nortel signing server perform the challenge–response sequence. IP client designers can access vxshell. The GNPS/PV/FS should work with a designer at any point they need access.

You can access two levels of diagnostic shell privileges: PDT and VxShell.

#### PDT level

The PDT level provides the full set of PDT commands. The prompt of PDT level is PDT>.

To access the PDT level, log on the IP Phone from an SSH client and enter the SSH user ID and SSH password. The IP Phone displays a banner, as shown in the following example.

```
Welcome to Nortel problem determination tool.
You are connected to IP Phone 1120E.
HW version:  31380016CA0081E86624
FW version 0625C39
MAC 0016CA0081E8 IP 47.128.38.163
Type “pdtHelp” for list of available commands.
Type “bye” to exit current shell.
```

```
PDT>
Table 100
PDT commands
Command Function
pdthelp Print pdt shell help
setLogLevel Set log level, Critical:  1, Major:  2, Minor:
3, Info:  4
setRecoveryLevel Set recovery level, Critical:  1, Major:
2, Minor:  3
Command Function
printLogLevel Print current log level, Critical:  1, Major:
2, Minor:  3, Info:  4
printRecoveryLevel Print current recovery level, Critical:
1, Major:  2, Minor:  3
printLogFile Print error log file
clearLogFile Clear content of error log file
taskMonShow Show task monitor list
taskMonAddTask Add a task to task monitor
taskMonRemoveTask Remove a task from task monitor
setCpuSamplingPeriod Set CPU sampling period, range:
30-120s, step 10s
printSetInfo Print hardware ID, firmware ID and MAC address
vxshell Switch into VxWorks Shell
bye Exit current shell
```

## VxWorks level

The VxShell level is a superuser level, which provides the VxWorks native shell (tShell) and operates in context. The prompt is the tShell prompt ->. You can access all VxWorks tShell commands and a full set of PDTshell commands.

To access the VxWorks shell, enter **`PDT> vxshell`**. To exit the VxShell, type **`bye`**. If you require authentication to access to the VxShell, the telephone shows a 22-digit random number in hexadecimal format. You must send this random number to the signing server. The signing server generates a 20-digit authentication key in hexadecimal format. You can manually send this key to the IP Phone for authentication. Copy and paste random number and the authentication key between the SSH client window and signing server window. If the two keys match, you can access the VxShell. The authentication key is based on a secret shared by the IP Phone and the signing server.

The following example shows command output.

```
PDT> vxshell

You need to pass the challenge question to get into vxshell.
```

```
The challenge number is: 012f2b1538613349565058 Please
get the response from the signing server:http://clientsig.
ca.nortel.com/sign/home.php

Enter the response: 97d9ca05a0ba99ad8279
-> bye
PDT>
Note: You must enter bye to exit the vxshell instead of
exit, which shuts down the tShell. If the tShell gets shut
down, you do not see the -> or the PDT shell prompt
PDT->. The IP Phone does not accept any commands at this
state. To return to the VxShell, press key [ctrl] [c] to
restart the tShell.
Table 101
VxWorks commands
Command Function
i Display all task info
ti Complete info on TCB for task
tt Task trace
memshow Show system memory partition blocks and statistics
chkstack Print a task stack usage
ls List contents of directory, -f : include details
lsr Recursive list of directory contents
cd Set current working path
ping Test that a remote host is reachable
traceroute Trace route to any host
netinfo Print common network info
routeshow Display host and network routing tables and stats
arpShow Display entries in the system ARP table
```

## Error logging

Log error information into the flash file system. The length of log file is 64 KB fixed and is written as a circular buffer.

You can issue a command to display or remove the log file on the console.

You can issue a command to configure or display the log level on the console. The log levels are the same as the severity levels. The default log level the telephone provides is Minor. The log level is stored in the provisioning file.

> ***Note:*** Because the length of the log file is 64 KB, use the Info level only if required to avoid overwriting the higher severity level records.

Each error is logged as a record. Each record has the same format regardless of which monitor generates it or the severity level and is organized into three sections. The first section is the mandatory information for each record including severity level, severity flag, time stamp, firmware version, source file information, error number, and a brief description, as shown in the following example.

```
=== Record #001 ===
MAJOR SET Logged 01/07/2002 00:34:35
Firmware:  06A5C1Hd10
File:  EcrTaskMonitor.c Line #505 Error #4
Description:  Task Monitor:  the Transport task is
suspended
```

The second section of the error record is an option. If the task is registered in the list of stack overflow events, the output is as shown in the following example.

```
ERROR*ecrStackShow::StackOverflow:  PDT
tpStackBase = 0x8194ffa0, pStackLimit=0x8194bfa0,
pStackEnd= 0x8194bfa0
tstack:  base 0x8194ffa0 end 0x8194bfa0 size 16368 high
1492 margin 14876
```

The third section of the error record is the supplementary information. The content depends on the flag in the calling function, as shown in the following example.

```
ECR_LOG_NO_EXTRA_INFO: no supplementary info
ECR_LOG_TASK_INFO: log task info (ti, tt, the stack
information from SP-96 to SP+96)
ECR_LOG_SUM_TASK_INFO: log summary of each task's TCB (i)
ECR_LOG_MEM_INFO: log memory usage info (memShow) Summary
info for all tasks:

NAME ENTRY TID PRI STATUS PC SP ERRNO DELAY
---------- ------------ -------- --- ---------- --------
-------- ------- -----
tExcTask excTask 81ff93d0 0 PEND 8078cc18 81ff92b0 3006b 0
tLogTask logTask 81ff6840 0 PEND 8078cc18 81ff6728 0 0
tNbioLog 805f84a0 81ff4130 0 PEND 80634554 81ff4078 0 0
DOS 80018f2c 81be2f80 0 DELAY 8060e8a8 81be2ef0 0 3
tShell shell 81bcba70 1 PEND 80634554 81bcb690 1c0001 0
VxTaskTerm VxTaskTermin 8199f290 2 PEND 80634554 8199f218 0
0
tWdbTask wdbTask 81bcdde0 3 PEND 80634554 81bcdb50 3d0002 0
RTPT 80064848 819c3040 8 PEND 80634554 819c2870 3d0002 0
SNDT 8066ba14 819c0c20 8 DELAY 8060e8a8 819c08c8 0 1
TimeSave 8067c0b4 819f8820 15 DELAY 8060e8a8 819f8798 0
3362
```

```
CpuMon 800ec5cc 81a1c690 19 DELAY 8060e8a8 81a1c5d0 0 160
hwtk 8051d994 819c8070 20 SUSPEND 80634554 819c7ff0 0 0
ECR_WDOG 800e977c 81a24ab0 49 PEND 80634554 81a24a38 0 0
TaskMon 800ebf48 81a208a0 49 READY 8060fb38 81a1fb30 1c0001
0
tNetTask netTask 81cc7770 50 READY 80634554 81cc76f8 0 0
NIRX 80018b8c 81be07c0 50 READY 80634554 81be05b0 b 0
tDhcpcState80536ccc 81bd9ee0 56 PEND 80634554 81bd9e40 31 0
tDhcpcReadTdhcpcRead 81bd88e0 56 PEND 80634554 81bd86c8
3d0002 0
tUglInput uglInputTask 81a9e290 60 READY 80634554 81a9df40
3d0004 0
dhcpTask sDHCPTask__5 819b5a20 60 PEND 80634554 819b58f8 0
0
t1 8064350c 81de8e30 100 PEND 8078cc18 81de8d00 0 0
usbOhciIsr 80655614 81ce4a40 100 PEND 80634554 81ce49d8 0 0
BusM A 80659590 81ce2790 100 READY 8060e8a8 81ce2700 0 0
usbMouseLib8064350c 81cdfda0 100 PEND 8078cc18 81cdfb40 0 0
usbMouseLib8064350c 81cdb470 100 PEND 8078cc18 81cdb378 0 0
usbKeyboard8064350c 81cd2810 100 PEND 8078cc18 81cd25b0 0 0
usbKeyboard8064350c 81ccdee0 100 PEND 8078cc18 81ccdde8 0 0
tTffsPTask flPollTask 81cc98d0 100 READY 8060e8a8 81cc9830
3d0002 0
Cursor 8006dec8 81a81640 100 READY 80634554 81a81578 3d0004
0
StickyTimer8007be40 81a37f30 100 READY 80634554 81a37e58
3d0004 0
EVTPROC 8019e810 81a2ccc0 100 READY 80634554 81a2cba0
3d0004 0
EtherSet 80194d68 819f7610 100 READY 80634554 819f72e0
3d0004 0
Link 80194d68 819e7400 100 READY 80634554 819e7310 3d0004 0
tDot1xSupp 80513550 819bdd60 100 PEND 8078cc18 819bdc68 0 0
RTCPSTS 801e10a8 819a56f0 100 PEND 8078cc18 819a55b0 3d0002
0
ProcSIPEvt 802327b0 819a0660 100 READY 80634554 819a0548
3d0004 0
Cdnr0 802157f4 819478b0 100 READY 80634554 819476e8 3d0004
0
DTMFTx0 802157f4 8193fd30 100 READY 80634554 8193fc48
3d0004 0
BLST 800d2310 81a36bb0 125 PEND+T 80634554 81a36b28 0 87194
DISR 8002187c 819e61f0 125 PEND 80634554 819e6168 0 0
FLASHICON 80021558 819d5fe0 125 PEND 80634554 819d5f60 0 0
INDR 800551a0 819d4ce0 125 PEND 80634554 819d4c58 0 0
HOOK 8005eec0 819d3a70 125 READY 8060e8a8 819d3998 0 0
KPD_CALLBAC8005f890 819d1320 125 PEND 80634554 819d1180 0 0
```

```
KBDR 8005f5d0 819cf0b0 125 PEND 80634554 819cf028 0 0
RTC 800536ac 819cde00 125 READY 80634554 819cdd48 0 0
CDT CDTUpdate 819ccb50 125 READY 80634554 819ccaa8 0 0
HDDET 8002f218 819cb880 125 READY 8060e8a8 819cb7b0 0 0
HSDET 8002f5c4 819ca5d0 125 PEND 80634554 819ca520 3d0004 0
EPTR eptReadThrea 819c9360 125 READY 80634554 819c9160
3d0004 0
CDNC 800463f8 81ffc2f0 125 PEND 80634554 81ffc238 0 0
SIGT 8003efdc 819c5460 125 READY 8060e8a8 819c5380 0 0
HAPIGET hapiGetTask 819c4250 125 READY 80634554 819c4120
3d0004 0
WAVT 800d3d24 819c1e30 125 PEND 80634554 819c1da8 0 0
i200xFullUp8008a1e8 81a86700 130 READY 80634554 81a86578
3d0004 0
i200xShortU800c8698 81a39250 130 READY 80634554 81a390c8
3d0004 0
SMC::TimerSThreadEntry_ 8199e010 150 PEND+T 80634554
8199dda8 3d0002 600
smce_app_thThreadEntry_ 8199bf30 150 PEND 80634554
8199bc08 3d0002 0
Transport tThreadEntry_ 81997320 150 PEND+T 80634554
81996dc0 3d0002 429491
327
UA thread ThreadEntry_ 81992620 150 PEND+T 80634554
81992440 3d0002 276492
DNS thread ThreadEntry_ 8198d900 150 PEND+T 80634554
8198d720 3006b 214743 6140
blog printQTask 81a35770 180 READY 8060e8a8 81a35680 0 0
dsp_assert 80520698 81fffdb0 180 PEND 80634554 81fffd30 0 0
winmgr winAppTask 81a6f9a0 200 PEND 8078cc18 81a6f7e8 0 0
i200xApp winAppTask 81a4cf80 200 PEND 8078cc18 81a4cdc8 0 0
Link 8001a2ec 819bace0 200 READY 8060e8a8 819bac28 0 0
ETHERSET_TI80194d68 819bfa10 201 READY 80634554 819bf928
3d0004 0
tAioIoTask0aioIoTask 81fe9ce0 250 PEND 80634554 81fe9c48 0
0
tDcacheUpd dcacheUpd 81aa4c90 250 READY 8060e8a8 81aa4bd8 0
0
tAioWait aioWaitTask 81ff0f50 251 PEND 80634554 81ff0e50
3d0002 0
Idle 800ec590 81a32e70 253 READY 8060e8a8 81a32de8 0 0
tUsbKbd 8064350c 81cd7140 255 READY 8060e8a8 81cd7088 0 0
ICPIDLE 8051dc20 81ffd640 255 READY 80785584 81ffd5e8 0 0

Memory Usage Info:
status bytes blocks avg block max block
------ ---------- --------- ---------- ----------
```

```
current
free 9498400 186 51066 9249120
alloc 7210640 4915 1467 -
cumulative
alloc 81327184 29445 2762 -

Detailed info for task ID 0x819C8070:

NAME ENTRY TID PRI STATUS PC SP ERRNO DELAY
---------- ------------ -------- --- ---------- --------
-------- ------- -----
hwtk 8051d994 819c8070 20 SUSPEND 80634554 819e1938 0 0

stack:  base 0x819c8070 end 0x819c6070 size 8176 high 1432
margin 6744
options:  0x4
VX_DEALLOC_STACK

VxWorks Events
--------------
Events Pended on :  Not Pended
Received Events :  0x0
Options :  N/A

$0 = 0 t0 = 0 s0 = 0 t8 = 0
at = 80d70000 t1 = 1000ff00 s1 = 0 t9 = 80e70000
v0 = 0 t2 = 80e97e74 s2 = 0 k0 = 0
v1 = 3fe t3 = 0 s3 = 0 k1 = 0
a0 = 50 t4 = 80e7e308 s4 = 0 gp = 80d94a50
a1 = 21 t5 = 82 s5 = 0 sp = 819c7ff0
a2 = 1 t6 = 203ac098 s6 = 0 s8 = 819c8010
a3 = 80efec72 t7 = 0 s7 = 0 ra = 807830fc
divlo = 6 divhi = 4 sr = 1000ff01 pc = 80634554

Task Trace:
819c8030 _pthread_setcanceltype+ac8a64:  KNL_RunReadyThr
eads (819c8280, ffffffff, 0, 0)
807830f4 KNL_RunReadyThreads+74 :  semQPut (&KNL_gGlobals
, 80782dac, 819c8000, 80 0ecb50)

stack dump from sp-96 to sp+96

819e18d0:  0000 0000 819a f200 * ......*
819e18e0:  0000 0000 8059 aa48 8039 3890 8086 1884
*.....Y.H.98.....*
819e18f0:  eeee eeee eeee eeee eeee eeee 0000 0000
*................*
```

```
819e1900:  819e 1908 8012 f7b8 81be 2f30 0000 8010
*........../0....*
819e1910:  819e 1978 819e 1958 803a 2894 819e 1958
*...x...X.:(....X*
819e1920:  8059 aa48 803a 0624 0000 0000 0000 0000
*.Y.H.:.$........*
819e1930:  0000 0000 819e 1958 81be 2f30 0000 8010
*.......X../0....*
819e1940:  819e 1978 0000 0000 803a 2894 819e 1968
*...x.....:(....h*
819e1950:  801f 07bc 0000 0000 819e 1c70 0000 0064
*...........p...d*
819e1960:  0000 0000 819e 1968 0000 0064 819e 1990
*.......h...d....*
819e1970:  819e 1978 801e f1f0 819e 9630 0000 03e8
*...x.......0....*
819e1980:  0000 0019 819e 19c0 819e 1990 801d fa58
*...............X*
819e1990:  819e 95f0 0000 1079 0000 0000 0000 0000
*.......y........*
819e19a0:  0000 0000 eeee eeee 0000 0001 805a 05dc
*.............Z..*
819e19b0:  0000 1079 eeee eeee 819e 19c0 801d fa10
*...y............*
819e19c0:  819e 95f0 eeee eeee eeee eeee eeee eeee
*................*
819e19d0:  819e 19d8 801f 08a0 *................*
value = 21 = 0x15

Remote Diagnostics Access
```

## Set up SSH user ID and password

You must set up the user ID and password before you can access the telephone. The telephone stores the user ID and password in the provisioning file. The length of user ID and password is 4 to 12 characters. When the user is in the edit mode to enter a password, the characters appear. Otherwise, the password is displayed as asterisks to block it from being read.

In the text UI, select Local Diagnostics > Advanced Diag Tools. Select the Config SSH check box. Enter a User ID and Password.

In the GUI, select Local Diagnostics > Advanced Diag Tools. For Config SSH?, enter Y. Enter a User ID and Password.

You can use a freeware SSH client such as PuTTY. On a PuTTY terminal keyboard, configure the backspace key as Ctrl+h.

## PDT level

The first level is the PDT level which is a general user level, provides the full set of PDT commands. The prompt of PDT level is PDT>.

To access the PDT level, log on to the telephone from a SSH client and enter the user ID and password. If the user ID and password are correct, you enter the PDT shell, as shown in the following example.

```
Welcome to Nortel problem determination tool.
You are connected to IP Phone 1120E.
HW version:  31380016CA0081E86624
FW version 0625C39
MAC 0016CA0081E8
IP 47.128.38.163
Type "pdtHelp" for list of available commands.
Type "bye" to exit current shell.
PDT>
To exit the PDT shell, type the "bye" command.
```

## VxShell Level

The second level is the VxShell level, which is a superuser level, provides the VxWorks native shell (tShell) and operates in context. The prompt is the tShell prompt "->". You can access all VxWorks tShell commands and the full set of PDT shell commands.

To access the VxShell level, enter vxshell on the PDT shell and proceed through an authentication process if the load is the production load. In the designer load, you need not to go through authentication. You can control a compile option if you must go through authentication in different loads.

If the authentication succeeds, you enter the VxWorks native shell. If the authentication fails, no further message appears and you stay at the PDT level.

On the VxShell, the global I/O is redirected to PTY so the user can see the debug printf output on the serial port console. However, nothing appears on the serial port console until you exit the VxShell at which point, the printf is restored to the serial console.

To exit the vxshell, enter the **`bye`** command.

The following example shows the output for accessing the VxShell.

```
PDT> vxshell
You need to pass the challenge question to get into vxshell.
The challenge number is:  012f2b1538613349565058
Please get the response from the signing serve:
```

```
http://clientsig.ca.nortel.com/sign/home.php
Enter the response:97d9ca05a0ba99ad8279
To exit the VxShell:
-> bye
```

## VxShell level privilege authentication process

To access the VxShell, enter **`vxshell`** in the PDT shell. The telephone presents a 22-digit random number in hexadecimal format. Send this random number to the signing server. The signing server generates a 20-digit authentication key in hexadecimal format. Manually send this key to the telephone for authentication. Typically, you must copy and paste the random seed and the authentication key between the SSH client window and signing server window. If the two keys match, you can access the VxShell. The authentication key is based on a secret shared by the telephone and the signing server.

The signing server is a Nortel proprietary application that can generate key s. The application is located at **clientsig.ca.nortel.com/sign/home.php**.

You must be granted permission from the IP clients team before attempting access to the signing server.

## PDT commands

The following table lists the PDT commands.

**Table 111**
**PDT commands**

| Command | Description |
|---|---|
| Feature specific commands | |
| pdtHelp | Print pdt shell help |
| setLogLevel | Set log level: Critical: 1, Major: 2, Minor: 3, Info: 4 |
| setRecoveryLevel <recovery level> | Set recovery level, Critical: 1, Major: 2, Minor: 3 |
| printLogLevel | Print current log level, Critical: 1, Major: 2, Minor: 3, Info: 4 |
| printRecoveryLevel | Print current recovery level, Critical: 1, Major: 2, Minor: 3 |
| printLogFile | Print error log file |
| clearLogFile | Clear content of error log file |
| taskMonShow | Show task monitor list |
| taskMonAddTask < taskName \| task id> | Add a task to task minitor |
| taskMonRemoveTask < taskName \| task id> | Remove a task from task monitor |

**Table 111**
**PDT commands (cont'd.)**

| Command | Description |
|---|---|
| setCpuSamplingPeriod <sampling period> | Set CPU sampling period, range: 180 - 360s, step 10s |
| listcerts | List all trusted certificates |
| printcert | Print a trusted certificate in detail |
| listcrls | Prints a detailed list of CRLs |
| listdevcerts | Print all device certificates |
| listsecuritylogs | List all events logged through the security interface |
| securitypolicy | Prints the current Security Policy values |
| gxasinfo | List the GXAS configuration and current status |
| printSetInfo | Print hardware ID, firmware ID and MAC address |
| vxshell | Switch to VxWorks shell |
| bye | Exit current shell |
| VxWorks commands | |
| i | Display all task information |
| ti | Complete information on TCB for task |
| tt | Task trace |
| memShow [level] | Show system memory partition blocks and statistics |
| ls [dirname] [-f] | List contents of directory, –f : include details |
| lsr [dirname] | Recursive list of directory contents |
| cd [dirname] | Change the current working path |
| ping <host ip> [# of pings] | Verify that a remote host is reachable |
| tracert <host ip> [max hops] | Trace route to a host |
| netinfo | Print common network information |
| routeshow | Display host and network routing tables and statistics |
| arpShow | Display entries in the system ARP table |

# Alarm errors on the VGMC, Signaling Server, and MGC

This section describes alarm errors on the Voice Media Gateway Card (VGMC), Signaling Server, and MGC.

## VGMC faceplate maintenance display codes

The VGMC maintenance display provides the diagnostic status of the card during power-up, the operational state when in service, and error information about the functional state of the card. Faceplate Maintenance Display Code lists the normal and fault codes.

During power-up, the card performs multiple self-tests, including an internal RAM test, ALU test, address mode test, boot ROM test, timer test, and external RAM test. If a test fails, the card enters a maintenance loop, and no further processing is possible. A failure appears on the display to indicate which test failed. For example, if the timer test fails, F:06 is displayed.

If any other tests fail (up to and including the EEPROM test), a message appears for three seconds. If more than one test fails, the message indicates the first failure. If you select verbose mode (by the test input PIN on the backplane), the three-second failure message does not appear.

If the maintenance display shows a persistent T:20 (indicating a VGMC or Signaling Server application software failure) after the card reset during a software download procedure, call Nortel technical support for assistance to download new software to the card.

The following table lists the faceplate maintenance display codes.

**Table 112**
**Faceplate maintenance display codes**

| | | **SMC** | **ITG-P** |
|---|---|---|---|
| T:01 | F:01 | XA Internal RAM Test Failed | XA Internal RAM Test Failed |

**Table 112**
**Faceplate maintenance display codes (cont'd.)**

| | | **SMC** | **ITG-P** |
|---|---|---|---|
| T:02 | F:02 | XA ALU Test Failed | XA ALU Test Failed |
| T:03 | F:03 | XA Address Mode Test Failed | XA Address Mode Test Failed |
| T:04 | F:04 | Watchdog Timer Test Failed | Watchdog Timer Test Failed |
| T:05 | F:05 | Flash Test Failed | Flash Test Failed |
| T:06 | F:06 | Timer Test Failed | Timer Test Failed |
| T:07 | F:07 | XA External RAM Test Failed | XA External RAM Test Failed |
| T:08 | F:08 | Dongle not Detected | Host DPRAM Test Failed |
| T:09 | F:09 | Time Switch FPGA failed | DS30 DPRAM Test Failed |
| T:10 | F:10 | ISPDI FPGA failed | Dongle Not Detected |
| T:11 | F:11 | Host DPRAM Test Failed | – |
| T:12 | F:12 | DS30X DPRAM Test failed | PCI LCA Failed |
| T:13 | F:13 | Serial EEPROM Test Failed | DS30X LCA Failed |
| T:14 | F:14 | – | CE-MUX LCA Failed |
| T:15 | F:15 | – | DSP LCA failed |
| T:16 | F:16 | – | Daughterboard LCA failed |
| T:17 | F:17 | – | CEMUX Register Test Failed |
| T:18 | F:18 | – | Serial EEPROM Test failed |
| T:19 | – | Waiting for application startup messages from processor. | |
| T:20 | – | CardLAN enabled, transmitting BOOTP requests. If this display persists, then the ITG Line card is running in BIOS ROM mode due to card software failure. | |

**Table 112**
**Faceplate maintenance display codes (cont'd.)**

| | | SMC | ITG-P |
|---|---|---|---|
| T:21 | – | CardLAN operational, A07 enabled, display now under host control. Card is looking for an active leader by sending BOOTP requests on the management LAN. A follower card sends BOOTP requests on the management LAN continuously. Enter +++ to escape from BOOTP request mode and start VGMC shell. | |
| T:22 | – | Attempting to start the application. | |
| T:23 | F:23 | IXP1200 Polling Failure 1 | Pentium Polling Failure 1 |
| T:24 | F:24 | IXP1200 Polling Failure 2 | Pentium Polling Failure 2 |
| Lxxx | – | Card is configured as the node leader card. xxx = number of IP Phones registered on the card | |
| Mxxx | – | Card is not configured as the Leader but is currently the node master. xxx = number of IP Phones registered on the card | |
| Fxxx | – | Card has detected the node master and is operating as a follower. xxx = number of IP Phones registered on the card. | |

## MGC four-character LED faceplate display

You can check the MGC four-character LED faceplate display for diagnostic information during bootup and normal operation.

### Use of the four-character LED display during MGC boot

When the system boots, the diagnostic information from the hardware and firmware sanity tests appears on the faceplate. The messages that appear on the four-character LED display during MGC bootup are shown in the following table. Further design investigation is required to confirm that all suggested messages during bootup are possible.

**Table 113**
**Messages displayed on the four-character LED display during MGC bootup**

| Message | Description |
|---|---|
| BOOT | This is the first message when display becomes active. |
| POST | Power on self test message appears while the MGC performs hardware system tests during system power up. |

**Table 113**
**Messages displayed on the four-character LED display during MGC bootup (cont'd.)**

| Message | Description |
|---|---|
| PASS | Power on self test pass. |
| EXXX | Error code where XXX is a numeric value. An error code appears if a serious system error is detected. |
| LOAD | Application software is loading. |

In normal operation the messages appear in the following order: BOOT, POST, PASS, LOAD. If a fatal self-test error occurs, then PASS and LOAD do not appear. Instead an error code appears.

## Use of the four character LED during MGC normal operation

After the boot loader loads the CSP and starts to run, you can use the faceplate display for diagnostic information.

During normal operation, the four-character LED displays the superloop and shelf number of the Media Gateway 1000E (MG 1000E) with MGC. If an error occurs that requires a diagnostic message to appear, then the display cycles between displaying the cabinet number and the error code. Each item appears for 20 seconds. The following table shows the messages that appear.

**Table 114**
**Messages displayed on the four character LED during MGC normal operation**

| Message | Description |
|---|---|
| EXXX | Error code where XXX is a numeric value. An error code appears if a serious system error is detected. |
| LLL$^S$ | IPMG super loop and shelf number where LLL is the superloop number and S is the shelf number. For example, 032$^0$ or 120$^1$.<br><br>***Note:*** Although this appears to be four characters, the superscript shelf digit is a special character [that is, 0 and 1] defined internally to the MGC to ensure it looks different from the loop designators. |
| E001 | A 96 port daughter board is installed in DB position 2, this is not supported. |

**Table 114**
**Messages displayed on the four character LED during MGC normal operation (cont’d.)**

| Message | Description |
|---|---|
| E002 | Unable to send registration request to call server. |
| E003 | Link down to call server. |
| E004 | Daughter board is not registered with call server.<br><br>Additional messages are defined for the VoIP DSP Daughter boards. |

# VGMC and Signaling Server configuration files

## Introduction

The active configuration files are in the /CONFIG directory and are named CONFIG.INI and BOOTP.TAB. OTM/EM creates these files and downloads them to the card. The BOOTP.TAB file contains the node properties, while the CONFIG.INI file contains the Card properties. With the introduction of the node level TN password feature in VGMC 2.2, a configuration file named SECURITY.INI contains the node password. The loss plan adjustment commands are in the LOSS.INI file.

Beginning in Release 4.0, all systems use a CONFIG.INI file that is centrally stored on the Call Server. The Signaling Server and VGMC access the Call Server by using the Call Server IP address stored in the BOOTP.TAB file (Leader) or in the BOOTP response (Follower). Therefore, the BOOTP.TAB file is also updated to have the Call Server IP address.

The system automatically backs up the configuration files to file names with .BAK extentions (for example, CONFIG.BAK and BOOTP.BAK). In case of a bad file download or file corruption, the system replaces the active copy with this version of the file.

An additional file, TPS.INI, can be created. This file is not created by OTM/EM but is a text file that you can use to control aspects of the VGMC application operation. When the application boots, if the file is in the C:/CONFIG directory, the contents are parsed and used in the application operation. If the file is not present, the application operates as normal. This file is typically used only for debugging and does not remain in the CONFIG directory.

Another file, UMS.INI, is created by the VGMC or Signaling Server application and stores the IP Phone firmware upgrade information.

> ***Note:*** Drive letters must be capitalized; file names can be lowercase. All files must follow the 8.3 format.

## Displaying contents

Print the contents of each file to the console by using the following command (the second parameter defaults to zero and remain off):

```
->copy "<path/filename>", 0
```

### Creating a file from CLI

In an extreme case where OTM/EM is not available or the LAN connection does not work, you can create a new file and write its contents from the terminal keyboard. The following command copies data from the standard input device (the keyboard) into the specified file. Everything typed goes directly into the file. End the input by pressing Ctrl+D at the start of a blank line. This closes the file and returns the VxWorks prompt. Copy the existing file to a backup file name (for example, **`-> copy "CONFIG.INI", "CONFIG.SAV"`**) and then copy the new file to the CONFIG.INI file name.

The following example is from the SMC, but the same information appears on the ITG-P or Signaling Server.

```
-> copy 0,"temp.ini"
Hello world!
^Dvalue = 0 = 0x0
-> copy "temp.ini"
Hello world!
value = 0 = 0x0
```

***Note:*** To avoid incorrectly configuring a node by typing invalid information, Nortel recommends that you do not use this procedure.

## Differences between configuration files

The various EM/OTM versions generate different CONFIG.INI and BOOTP.TAB files. If you move a VGMC from a Meridian 1 system to a CS 1000 or CS 1000M system or vice versa, be sure to download or transfer the card and node properties so the CONFIG.INI and BOOTP.TAB files are updated to the correct format.

## File details

### CONFIG.INI

The MAM task parses this file to configure the routes, the DSP codecs and parameters, Call Server ELAN IP address (active ELNK), SNMP traps, DSCP bits, and 802.1Q priority bits. The file is fully parsed on card reset. It is also parsed when the file is transmitted by OTM or transferred by EM. In these cases, the application parses some sections of the CONFIG.INI file and modifies the prior VGMC operation so no card reboot is required for data changed in those areas. Changes in other areas of the file (for example, the [TlanConfig] section) still require a card reboot before the changes can take effect. You can change the following sections without performing a reboot:

- [snmp]
- [routes]
- [dsp0]
- [tos]
- [ElanConfig]
- [Loss Plan]
- [firmware]

You can download the file through the VGMC shell by using the command configFileGet.

Starting in Release 4.0, the following line is examined and compared between the copy of the CONFIG.IN file stored on the Call Server and the copy stored locally on the Signaling Server or VGMC:

```
#dateOfLastFileModification(dd/mm/yyyy hh:mm:ss)=21/08/2
003 01:45:12 i.
```

If the line is different, then the CONFIG.INI file on the Call Server is used.

The configuration file is self-explanatory. Each section begins with a section identifier in brackets ([]).

> ***Note:*** The [routes] section is for creating static route entries on the ELAN interface only. The entry includes the destination IP address and the subnet mask.

## EM file

The following is an example of a CONFIG.INI file generated by the Succession 4.0 Element Manager.

```
#version=VGMC4.0
#authoringApplication=EM
#dateOfLastFileModification(dd/mm/yyyy hh:mm:ss)=21/08/2
003 01:45:12

[keycode]
keycodeId=12345678-12345678-12345678

[Security]
ElanAccessOnly=0

[snmp]
```

```
rdCommunityName=public
wrCommunityName=private
trapsEnabled=1
trapsub=0.0.0.0
ts1=0.0.0.0,255.255.254.0,
CardIP=47.11.255.26
sysHostName=
sysLocation=
sysContact=
CardIP=47.11.255.33
sysHostName=
sysLocation=
sysContact=

[routes]
re1=0.0.0.0,255.255.254.0,

[dsp0]
EchoCancel=1
DspEchoTail=128
VadThreshold=-17
IdleNoise=-65
DtmfToneDetect=1
ModemDetect=1
FaxDetect=1
FaxRate=9600
FaxPlayoutNomD=100
FaxActiveTimeout=200
FaxPacketSize=30
Codec=1
VxPayload=30
VxPlayoutNomD=60
VxPlayoutMaxD=120
VadEnabled=0
Codec=5
VxPayload=20
VxPlayoutNomD=40
VxPlayoutMaxD=80
VadEnabled=0
Codec=2
VxPayload=30
VxPlayoutNomD=60
VxPlayoutMaxD=120
VadEnabled=0
Codec=9
VxPayload=30
VxPlayoutNomD=60
```

```
VxPlayoutMaxD=120
VadEnabled=0
Codec=8
VxPayload=1
VxPlayoutNomD=60
VxPlayoutMaxD=120
VadEnabled=0

[tos]
controlPrio=160
voicePrio=184
802.1pqEnabled=0
802.1p=6
802.1q=0
natEnabled=0
natTimeout=90

[ElanConfig]
CallServerIP=47.11.255.0
SurvivalIP=47.11.254.193
SignalPort=15000
BroadcastPort=15001

[TlanConfig]
SignalPort=5000
AudioPort=5200

[GateKeeper]
PrimaryGKIP=47.11.249.140
AlternateGKIP=47.11.249.106
PrimaryNCSIP=47.11.249.140
PrimaryNCSPort=16500
AlternateNCSIP=47.11.249.106
AlternateNCSPort=16500
NCSTimeout=10

[firmware]
serverIP=0.0.0.0
subnetMask=255.255.254.0
fwfileDirPath=download/firmware/
userID=
password=

[ApplicationServer_47.11.255.26]
HostName=
H323ID=bvw_lab_node_5
SW_VtrkTPS=1
```

```
SW_GateKeeper=0
SW_SetTPS=1

[ApplicationServer_47.11.255.33]
HostName=
H323ID=bvw_lab_node_5
SW_VtrkTPS=0
SW_GateKeeper=0
SW_SetTPS=1

[SNTP Server]
Mode=active
Interval=256
Port=20555

[SNTP Client]
Mode=passive
Interval=256
Port=20555
ServerIP=0.0.0.0

[OM Thresholds]
PacketLoss=10
Latency=250
Jitter=30
PollingPeriod=5
CallServerReporting=1
```

The network time client/server moved from the TPS.INI file to the EM created CONFIG.INI file. The following parameters are available:

- Mode — defines the mode of the server operation as follows:
  - active — sends broadcast of time at defined intervals.
  - passive — waits for query from clients (default)
  - disabled — time server not started
- Interval — assigns the interval between time updates and can be set to any 2n value (31 > n > 0) (256)
- Port — assigns the UDP port used by the time client/server (20,000 + node ID)

## OTM2.1 file

The following is an example of a CONFIG.INI file generated by OTM 2.2.

```
#version=VGMC4.0
#authoringApplication=OTM
#dateOfLastFileModification(dd/mm/yyyy hh:mm:ss)=04/11/2
003 10:40:10

[keycode]
keycodeId=12345678-12345678-12345678

[snmp]
rdCommunityName=public
wrCommunityName=private
trapsEnabled=1
trapsub=47.11.243.38,47.11.243.206
ts1=47.11.243.38,255.255.255.0,
ts2=47.11.243.206,255.255.255.0,
CardIP=47.11.155.135
sysHostName=vf5-main
sysLocation=M1 lab
sysContact=Yvonne Zhu
CardIP=47.11.155.136
sysHostName=vf5-main
sysLocation=m1 lab
sysContact=yvonne zhu *

[routes]

[dsp0]
EchoCancel=1
DspEchoTail=128
VadThreshold=-17
IdleNoise=-65
DtmfToneDetect=1
ModemDetect=1
FaxDetect=1
FaxRate=14400
FaxPlayoutNomD=100
FaxActiveTimeout=20
FaxPacketSize=30
Codec=1
VxPayload=10
VxPlayoutNomD=20
VxPlayoutMaxD=40
VadEnabled=0
Codec=8
VxPayload=1
VxPlayoutNomD=40
VxPlayoutMaxD=80
```

```
VadEnabled=0
Codec=9
VxPayload=30
VxPlayoutNomD=60
VxPlayoutMaxD=120
VadEnabled=0

[tos]
controlPrio=160
voicePrio=184
802.1pqEnabled=1
802.1p=6
802.1q=0
natEnabled=1
natTimeout=90

[ElanConfig]
CallServerIP=47.11.155.158
SurvivalIP=47.11.155.133
SignalPort=15000
BroadcastPort=15001

[TlanConfig]
SignalPort=5000
AudioPort=5200

[firmware]
serverIP=47.11.225.53
subnetMask=255.255.255.0
fwfileDirPath=/auto/edavey/I2004_FIRMWARE/REL58/
userID=edavey
password=har8old

[SNTP Server]
Mode=active
Interval=256
Port=21134

[SNTP Client]
Mode=passive
Interval=256
Port=21134
ServerIP=47.11.155.158

[security]
ElanAccessOnly=0
```

```
[OM Thresholds]
PacketLoss=10
Latency=250
Jitter=30
PollingPeriod=5
CallServerReporting=1
->
```

## BOOTP.TAB

The BOOTP.TAB file is present on all VGMC and Signaling Servers in the node. This file contains the node properties. The file is fully parsed on card reset. It is also parsed when the file is transmitted by OTM or transferred by EM. In these cases, the application's run-time configuration enhancement parses the BOOTP.TAB file and allows card additions or deletions with no card reboot required. Changes in other areas of the file (for example, card IP addresses or subnet masks) still require a card reboot before taking effect.

In Release 4.0, the Call Server IP address was added to the BOOTP.TAB file to allow the central distribution of the CONFIG.INI file from the Call Server to all VGMC and Signaling Servers in the node.

> ***Note:*** Even though the first lines appear as comments within the BOOTP.TAB file, the administrative applications and the VGMC application use them; do not modify them. Comments elsewhere within the BOOTP.TAB file (designated by a leading number sign [#]) are ignored by the BOOTP server.

### Differences between the EM and OTM2.0 file

Differences between the BOOTP.TAB file formats are as follows:

- The OTM2.2 version does not specify a card role, while the EM version does, as shown in the following examples.

  In OTM, ITG-P card, no role specified:
  `#WEB_MGMT 47.11.242.193 P"`
  In EM, ITG-P card, follower `#WEB_MGMT 47.11.216.186 P Follower`

- In the EM version, an extra field is specified:
  `hd="/u/config":`

### EM file

The following example is the BOOTP.TAB file generated by EM:

```
#192.168.20.10
#47.11.216.187
#version=VGMCINE4.0
#WEB_MGMT 47.11.216.187 SS Leader
#WEB_MGMT 47.11.216.186 P Follower
#WEB_MGMT 47.11.216.229 P Follower
#WEB_MGMT 47.11.216.230 SA Follower

.subnet1
:sm=255.255.254.0:gw=47.11.216.1:ts=192.168.20.10:hn:cs
=47.11.155.158:

1:tc=.subnet1:ha="00:02:b3:86:29:3a":ip=47.11.216.187:l
p=192.168.20.2 255.255.0.0 192.168.20.1:dn=:to=7812:hd=
"/u/config":
2:tc=.subnet1:ha="00:60:38:8e:12:00":ip=47.11.216
.186:lp=192.168.20.3 255.255.0.0 192.168.20.1:dn=4
0:to=7812:hd="/C:/config":
3:tc=.subnet1:ha="00:60:38:8e:12:99":ip=47.11.216
.229:lp=192.168.20.4 255.255.0.0 192.168.20.1:dn=8
0:to=7812:hd="/C:/config":
4:tc=.subnet1:ha="00:60:38:bd:b3:15":ip=47.11.216
.230:lp=192.168.20.5 255.255.0.0 192.168.20.1:dn=6
0:to=7812:hd="/C:/config":
```

## OTM2.2 file

The following BOOTP.TAB file was generated by OTM2.0:

```
#47.11.243.174
#47.11.242.193
#version=VGMC4.0
#WEB_MGMT 47.11.242.193 P
#WEB_MGMT 47.11.242.194 SA

.subnet1
:sm=255.255.255.0:gw=47.11.242.1:ts=47.11.243.174:hn:cs
=47.11.155.158:

1:tc=.subnet1:ha="00:60:38:01:3A:84":ip=47.11.242.193:l
p=47.11.243.175 255.255.255.0 47.11.243.1:dn=18 0:to=11:
2:tc=.subnet1:ha="00:60:38:BD:B3:53":ip=47.11.242.194:l
p=47.11.243.176 255.255.255.0 47.11.243.1:dn=20 0:to=11:
value = 0 = 0x0
```

## SECURITY.INI

The SECURITY.INI file exists if you enter a node-level TN password. The variable Security Check denotes whether the password is enabled or disabled. The variable Password displays the current password. This information is available through the command,nodePwdShow.

```
[Node Info]
SecurityCheck=1
PassWord=1234567890
```

## LOSS.INI

The LOSS.INI file exists if you enter either the lossPlanSet or UKLossPlanSet command. The file contains the RLR/SLR offset values that are downloaded to the IP Phones. Positive numbers are loss, negative values are gain.

If all values are 0 (for example, after you enter the lossPlanClr or UKL command), the application deletes the file. To view the file contents, enter the lossPlanPrt command.

The following example shows the file contents after you enter the UKLossPlanSet command.

```
-> copy "loss.ini"
[Loss Plan]
HandsetRLROffset=0
HandsetSLROffset=-5
HeadsetRLROffset=0
HeadsetSLROffset=-5
HandsfreeRLROffset=0
HandsfreeSLROffset=0
```

## TPS.INI

The TPS.INI file is an optional text file in the /CONFIG directory that you can use to modify the default (normal) behaviour of the LTPS. The format of the file is similar to the CONFIG.INI file: each section begins with a section header in brackets followed by a list of data items, each equated to some value.

The file is parsed on card bootup. If any parameters change, reboot the card so the changes can take effect.

> ***Note:*** Nortel does not recommend that you change any parameter unless technology or GNTS staff direct you to do so.

## Call Server link control

To disable the use of the TCP link between the VGMC or Signaling Server and the Call Server, assign a value of 0 (false) to the UseTcpLink= variable. The default is 1 (true) to use TCP signaling.

```
[pbxLib]
UseTcpLink=
```

To change the time between attempts to establish the link with the Call Server, add the following line to the TPS.INI file. The default is 15 seconds.

```
[pbxLib]
LinkResetTimer=
```

## RUDP windowing control

The IP Phone RUDP connections use message windowing. The default size is 10 outstanding (non-ACKed) messages. You can adjust the window size by adding the following lines in the TPS.INI file:

```
[usiLib]
rudpWindowSize=
```

## DSP gain limit control

The VGMC gateway normally limits the gain adjustment for the DSPs to +/– 6 dB. You can adjust this range by adding the following lines to the file:

```
[Loss Plan]
DspMaxLoss=
DspMinLoss=
```

## DSP echo canceller control

You can configure the echo canceller parameters for all channels. You can place the same variables under the heading [dsp0] in the CONFIG.INI file. However, values in the CONFIG.INI file are superseded by values in the TPS.INI file. If values neither appear in the TPS.INI or CONFIG.INI file, then the default values are used.

```
[Dsp Echo]
DspEchoOpcode=
DspEchoNlpFix=
DspEchoConfig=
DspEchoNlpAggress=
DspEchoCnConfig=
```

## IP parameter control

Adjust the timing of the IP Phone jitter buffer resynchronization mechanism by adding the following lines. The parameter earlyResync has a default value of 2000 milliseconds; lateResync has a default of 60000 milliseconds.

```
[IP Params]
earlyResync=
lateResync=
```

## TPS control

Modify the parameters of the election process for the VGMC node master by adding the following lines to the TPS.INI file. The default for each parameter appears in parenthesis.

ElectionDuration controls the length of time to wait for responses to an election request (2 seconds).

ElectionResult defines how often the Master notifies the other cards in the node of its existence (30 seconds).

ElectionTerm defines how long the nonmasters wait to obtain the Master broadcast (35 seconds).

Port defines the UDP port for the election communications (16543).

MaxTerminals defines the maximum number of terminals allowed to register on the card (by default: ITG-P: 96; SMC with 32 channels: 128; SMC with 8 channels: 32). Change the MaxTerminals value to change when the card notifies the Master that it cannot accept another IP Phone registration. Use care when you change this value. You cannot use one value across a node with mixed card types unless you want to restrict all cards to the same capacity.

```
[TPS Node]
ElectionDuration=
ElectionResult=
ElectionTerm=
Port=
MaxTerminals=
```

## Keymap download control

You can add two parameters to the TPS.INI file to modify the timing of the keymap download process. Both parameters are used in a calculation that determines how quickly requests are sent to the CS 1000 for keymap downloads. The default for each parameter appears in parentheses.

RTTExp is the expected round trip time between sending a keymap download request to the CS 1000 and receiving its response. When the RTT exceeds this value, the CS 1000 is busy so the equation introduces negative feedback to slow down the requests. This prevents the CS 1000 from losing requests due to buffer overflows (300 milliseconds).

Delay Factor is a multiplication factor used in the calculation (10).

```
[Keymaps]
RTTExp=
Delay Factor=
```

## Startup script

Sometimes you must ensure that certain commands run if a card reboots. You can place these commands in a startup script. This file must have the following path and file name, to be parsed by the VGMC:

"/C:/etc/startup"

If the "etc" directory is not present, create it by using the command mkdir.

Any command you can enter at the CLI, you can enter in the startup script. A script can be useful to debug problems, for example, if you want to automatically load object files. The following is an example of the contents of a startup script that adds a host and modifies the default gateway to be the ELAN instead of the TLAN.

```
-> copy "startup"
hostAdd("joeuser","47.11.229.34")
netDevCreate("joeuser:","47.11.229.34",1)
iam "nilantha","joeuserPassword"
routeDelete "0.0.0.0","10.1.1.1"
routeAdd "0.0.0.0","47.11.214.1"
value = 0 = 0x0
```

The startup script runs immediately after the vxWorks banner appears.

```
Adding 6239 symbols for standalone.
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
]]]]]]]]]]] ]]]] ]]]]]]]]]] ]] ]]]] (R)
] ]]]]]]]]] ]]]]]] ]]]]]]]] ]] ]]]]
]] ]]]]]]] ]]]]]]]] ]]]]]] ] ]] ]]]]
]]] ]]]]] ] ]]] ] ]]]] ]]] ]]]]]]]]] ]]]] ]] ]]]] ]] ]]]]]
]]]] ]]] ]] ] ]]] ]] ]]]]] ]]]]]] ]] ]]]]]]] ]]]] ]] ]]]]
]]]]] ] ]]]] ]]]]] ]]]]]]]] ]]]] ]] ]]]] ]]]]]]] ]]]]
```

```
]]]]]] ]]]]] ]]]]]] ] ]]]]] ]]]] ]] ]]]] ]]]]]]]] ]]]]
]]]]]]] ]]]]] ] ]]]]]] ] ]]] ]]]] ]] ]]]] ]]]] ]]]] ]]]]
]]]]]]]] ]]]]] ]]] ]]]]]]] ] ]]]]]]] ]]]] ]]]] ]]]] ]]]]]
]]]]]]]]]]]]]]]]]]]]]]]]]]]]]]
]]]]]]]]]]]]]]]]]]]]]]]]]]]]] Development System
]]]]]]]]]]]]]]]]]]]]]]]]]]]]
]]]]]]]]]]]]]]]]]]]]]]]]]]] VxWorks version 5.3.1
]]]]]]]]]]]]]]]]]]]]]]]]]] KERNEL: WIND version 2.5
]]]]]]]]]]]]]]]]]]]]]]]]] Copyright Wind River Systems,
Inc., 1984-1997

CPU: VPS Pentium MMC2.  Processor #0.
Memory Size:  0x2000000.  BSP version ITGL 2.10.63.2.

Executing startup script /C:/etc/startup ...
hostAdd("joeuser","47.11.229.34")
value = 0 = 0x0
netDevCreate("joeuser:","47.11.229.34",1)
value = 0 = 0x0
iam "joeuser","joeuserPassword"
value = 0 = 0x0
routeDelete "0.0.0.0","10.1.1.1"
value = 0 = 0x0
routeAdd "0.0.0.0","47.11.214.1"
value = 0 = 0x0
Done executing startup script /C:/etc/startup
```

# VGMC/Signaling Server/IP Phones installation and upgrades

## Installation overview

The Communication Server 1000 (CS 1000) documentation describes the commands and the process to install a new Voice Media Gateway Card (VGMC) or Signaling Server. In general, use the following procedure.

1. Set up the ELAN subnet mask, IP address, and gateway IP address. Configure the Leader flag and a flag to retrieve data from a local BOOTP table in the NVRAM of the node Leader 0 card. This enables you to use OTM/EM to use the ELAN interface for downloads or from the BIOS/VxWorks shell CLI for file transfers. Perform this only for the Leader; all remaining cards in the node retrieve IP data from the Leader through the BOOTP request–response mechanism. This step (and step 2) requires you to connect a terminal directly to the Leader card, either by using the faceplate or the octopus cable connection.
2. Download the application software to the VGMC/Signaling Server.
3. Download the node and Card properties from OTM/EM to the card. The node properties are downloaded in the BOOTP.TAB file. The Card properties are downloaded in the CONFIG.INI file.
4. Download the IP Phone firmware file to the VGMC/Signaling Server.
5. If the card does not have the current firmware version loaded, upgrade the card firmware using the xaUpgrade command.

### FTP Server and File Transfer Options

Normally, you can download the VGMC/Signaling Server application software, configuration files, and IP Phone firmware from the OTM PC or through EM. The VGMC/Signaling Server application does not support PPP, so you must have either a PC card in the faceplate or access to a device on the ELAN from which to provide a new load or to update the configuration files.

### The Meridian 1 or CSE 1000 system

Create a directory on the Call Server (for example, c:/u/itg) and place the VGMC/Signaling Server files in it. You can then either remotely transfer the files, using PPP, through the Call Server to this directory, or use FTP to copy the files to it when you have access to the OTM PC. You can then direct the swDownload function to that directory for the VGMC software upgrade. Likewise, you can the CONFIG.INI and BOOTP.TAB files to the cards using FTP. For the IP Phone firmware, initially transfer the compressed version of the file, using FTP, from a VGMC /FW directory to the Call Server. You can then transfer the files using FTP to any card that requires them in the future.

> ***Note:*** If you use the Call Server drive, ensure that the VGMC files do not cause the disk to become full, thus affecting Call Server operation.

### The CS 1000 or CS 1000M system

On the CS 1000 or CS 1000M system, all necessary files are stored on the Call Server or the Signaling Server disk. You can manually download the files from these platforms to the VGMC that requires them.

The Call Server has the CONFIG.INI and BOOTP.TAB files for each node on the system stored in the /u/db/node directory. You can manually retrieve the files from the CS 1000 or CS 1000M by using the bootPFileGet or configFileGet command from the VGMC CLI and specifying the path and file name to retrieve. Alternatively, you can the files using FTP from the Call Server, rename the files and then transfer the files to the card using FTP. The files for each node are differentiated by having the node ID as part of the file name. The file names follow the convention:

CONFIG.INI: nodeNNN.cfg
BOOTP.TAB: nodeNNN.btp
where NNN = node ID

The Signaling Server has the VGMC application software and the IP Phone firmware files. These are stored in the /u/fw directory.

Useful commands for transferring the files are as follows:

- **`swDownload`**
  Transfer, using FTP, the ITG application to the card.
- **`bootPFileGet`**
  Transfer, using FTP,using FTP,the BOOTP.TAB file to the card.
- **`configFileGet`**
  Transfer, using FTP, the CONFIG.INI file to the card.
- **`hostFileGet`**
  Transfer, using FTP, the specified file to the card.

- **`bootPFilePut`**
  Send the BOOTP.TAB file, using FTP, to the specified host.
- **`configFilePut`**
  Send the CONFIG.INI file, using FTP, to the specified host.
- **`hostFilePut`**
  Send the specified file, using FTP, to the specified host.

### Setting NVRAM Parameters from the VGMC/Signaling Server CLI

Configure the necessary NVRAM parameters for the Leader card by using the command setLeader from the VGMC CLI. Run this command when you install a card that already has VGMC software. When the card boots up, press the plus sign three times (+++) dots (....) print during the BOOTP query time. The card loads the existing application. Log on and enter the command.

On the Signaling Server, use the setup utility to configure this information.

## VGMC/Signaling Server upgrade overview

Typically, you perform upgrades by using the OTM software download or EM transfer function. However, you can initiate an upgrade from the VGMC CLI if required.

The commands differ depending on whether you perform them from the VGMC CLI prompt, the VxWorks CLI prompt, or the BIOS interface. The basic download sequence is as follows:

1. Erase the program area of flash memory to remove any old versions of the software from the card. On the ITG-P, the VGMC application saves the a copy of the current software when you download a new one from OTM/EM. If the new load fails to boot, the old software is used; this increases the card bootup time because the checksum must be calculated for both binaries. Erasing the flash speeds up the reboot time because after the download, only one binary is in flash. No copy of the software exists on the SMC card; only one C:/EXEC file exists on the card.
2. If necessary, prepare the FTP server or card for the download.
3. If necessary, set up the IP address information on the card.
4. Execute the download command to copy the software from the server by using an FTP session.
5. Reboot the VGMC to activate the new load.

## Software upgrades from the VGMC CLI

1. On the ITG-P card, erase the program storage area (see "upgradeErase" (page 287)).
   Skip this step for the SMC.
2. Set up the FTP server (optional). You must know the IP address of the server, the file name, and the path name. If you copy the file from the OTM PC, open the transmit dialog box (synchronize > transmit), select the card software check box, browse to the file to download, and click Open. Leave the transmit dialog box open; the FTP server is ready for the FTP request from the VGMC.
3. If necessary, configure the ELAN IP data by entering the command setLeader.
4. Initiate the software download by entering swDownload.
5. Reset the VGMC. Use this command when an IP Phone user hears a problem such as echo.

## Software upgrades from the VxWorks CLI

1. Erase the program storage area by entering the upgradeErase command.
   Skip this step for the SMC.
2. Set up the FTP server (optional).
   For more information about this step, see "Software upgrades from the VGMC CLI" (page 426).
3. If necessary, configure the ELAN IP data by entering the command setLeader.
4. Initiate the software download. The syntax varies depending on the location of the file.
   File is on another VGMC, Signaling Server, OTM PC or a network FTP server:
   ```
   swDownload
   ```
   If the file is on the ITG-P A: drive (PC card in faceplate):
   ```
   -> upgradePCMCIA "/A:filename"
   ```
   If the file is on the SMC A: drive (PC card in faceplate):
   ```
   -> copy "/A:/filename", "/C:EXEC"
   ```
5. Reset the VGMC. Use this command when an IP Phone user hears a problem such as echo.

   ***Note:*** On the SMC, remove the PC Card (if used) from the faceplate before you reset the card.

### Software upgrades from the VGMC BIOS

If no application is on the card (for example, card reboots after you run the ITG-P upgradeErase command but before you complete a download, or the SMC /C:/EXEC file is erased), when another application is on the card, or there a problem prevents the card from starting from the application image stored in flash, you can boot in to the BIOS and download a new version of software.

1. Erase the program storage area by entering the upgradeErase command.
   Skip this step for the SMC.
2. Set up the FTP server (optional).
   For more information about this step, see "Software upgrades from the VGMC CLI" (page 426).
3. Initiate the software download. The syntax varies depending on the location of the file.
   If the file is on the VGMC C: drive, Signaling Server, OTM PC or a network FTP server:
   `swDownload`
   If the file is on the ITG-P A: drive (PC card in faceplate):
   `BIOS> swCopy "/A:filename"`
   If the file is on the SMC A: drive (PC card in faceplate):
   `-> copy "/A:/filename", "/C:EXEC"`

   ***Note:*** The file name on the SMC C: drive must be EXEC.

4. Reset the VGMC:
   `BIOS> reboot (ITG-P)`

   or

   `-> sysReboot (SMC)`

   ***Note:*** On the SMC, remove the PC Card (if used) from the faceplate before you reset the card.

If you cannot access the OTM PC or the remote FTP server, you can use the ping command to debug the access (use the syntax described in "ping" (page 221).

You can also transfer the files to the card using FTP.

## VGMC/Signaling Server transmit node and card properties

You must download the node and card properties from OTM/EM to the new card. The Leader uses the node properties file (BOOTP.TAB) to configure its TLAN IP address and all IP address information on the other VGMC Follower cards in the node (see "BOOTP.TAB" (page 415)). OTM/EM downloads the file to the Leader, which transfers, using FTP, a copy of the file to the currently registered Followers.

Each card uses the card properties (CONFIG.INI) to configure codecs (see "CONFIG.INI" (page 408)). A card obtains the TLAN IP address from the BOOTP.TAB (usually indirectly through a BOOTP response) and the Call Server ELAN IP address from the CONFIG.INI file. Therefore, the link to the Call Server cannot be established unless the CONFIG.INI file is transmitted to the card.

You can change certain properties (such as codec parameters) without rebooting the card. However, you must reboot the card to read the CONFIG.INI file and process all changes (such as the Call Server IP address).

### Trick for resetting file lock in OTM

If the VGMC terminates during a file transmission, it may leave the file locked on the card. When a file is locked, you cannot transmit files.

The following procedure works only for the Leader card. When you transmit the node properties from OTM, a dialog box appears prompting you to override the file lock. If you confirm override, the file lock flag is cleared and the node properties are transmitted. Thereafter you can transmit the card properties if you were previously blocked due to the file lock being set already.

## Transmit IP Phone firmware

You can download the IP Phone firmware from OTM/EM to the VGMC/Signaling Server application to upgrade the IP Phone firmware. You can perform the download at any time to a card that had an established link with the Call Server but not had the C: drive reformatted. The VGMC/Signaling Server application creates the /FW directory the first time after the link to the Call Server is established. The directory is erased along with the other directories when the /C: drive is reformatted. In normal circumstances, the directory exists when you perform the download. If the download fails, check that the directory is present on the card (for example, change to the /C: directory; ll shows the firmware, or use the pbxLinkShow command to check the Call Server link status).

### Default firmware location

The default firmware locations are as follows:

- VGMC: /C:/fw
- Signaling Server: /u/fw

If no external server is configured or the VGMC has no firmware on the A: drive PC card, then during the VGMC application bootup, it takes the firmware file from this default location.

The files are stored compressed on the VGMC, so on those cards the files are then uncompressed and copied to the /ums directory in memory.

The files are not stored compressed on the Signaling Server, so on that card the files are copied to the /ums directory in memory.

## Default firmware file name

The default firmware file name for IP Phone 2004 is i2004.fw and for IP Phone 2002 is i2002.fw. This default file name is used when the VGMC application attempts to download firmware from the external server or from the local default location at bootup time.

## Valid firmware file name for download

When you download firmware from OTM/EM, the firmware file name must be in a defined format. Otherwise, the file is not recognized and the download fails. The valid format is as follows:

aabbcdd.bin

where aa = location code, bb = Terminal Type, c = firmware release, dd = firmware version

## Transmit IP Phone firmware to VGMCs

You can download the IP Phone firmware from OTM/EM to the VGMCs at any time. The VGMC application software compresses the firmware file as it is downloaded, writing the compressed file to the default firmware location. After the download finishes, the compressed firmware file is uncompressed and placed in the /ums directory in memory. The firmware policy is updated automatically so that you can perform IP Phone upgrades at any time.

If the file is transferred to the card by another means, such as hostFileGet or an external FTP application, this compression is bypassed, and the file uses significantly more disk space (for example, 1.85 MB versus 0.83 MB). Furthermore, do not use the copy command to transfer the firmware to the VGMC, as it takes longer and can corrupt the C: drive. Nortel recommends that you use an FTP transfer.

> ***Note:*** For the recommended method to manually transfer the firmware files without using OTM/EM, see “Transferring Firmware without OTM/EM” (page 430). If you manually download the firmware to the VGMC using another method, you must either issue the umsUpdatePolicy command on all cards or reboot all cards.

Transfers from OTM use the OTM FTP server user name and password. You can download to the VGMC even if the card was never connected to the Call Server.

On the other hand, EM uses the Call Server user name and password for the FTP transfer. If a card was never connected to the Call Server, the password synchronization feature did not download the Call Server user name and password to the card to be saved in the card NVRAM. It is unlikely that the default user name and password for the card match the Call Server logon, unless you manually match the card logon user name and password to the Call Server. Then, it is not possible to download the firmware files from EM until after the card connects to the Call Server.

## Transmit IP Phone firmware to Signaling Server

You can download the IP Phone firmware from EM to the Signaling Server at any time. The VGMC application software downloads the file from the EM station without compressing it. This uncompressed firmware file is placed in the default firmware location. After the download is completed, a copy of the firmware file is created in the directory named with the appropriate default firmware file name (to ensure the download to the VGMCs works). Then, a copy is placed in the /ums directory. The firmware policy is updated automatically so that you can perform IP Phone upgrades at any time.

## Transferring Firmware without OTM/EM

If the OTM/EM is not present, you can download the firmware by issuing the command firmwareFileGetI2004 or firmwareFileGetI2002. According to the input firmware file name, it downloads the appropriate file and updates the corresponding firmware policy. The commands are used as follows:

```
firmwareFileGetI2004 "hostIP", "uid", "password", "path",
"filename"
```

The following is an example for IP Phone 2004 firmware (file name = 0602B39.BIN).

```
firmwareFileGetI2004 "47.11.18.2", "etherset",
"etherset", "/itg/firmware/R1.39", "0602B39.BIN"
```

The following is an example for IP Phone 2002 firmware (file name = 0603B39.BIN).

```
firmwareFileGetI2002 "47.11.18.2", "etherset",
"etherset", "/itg/firmware/R1.39", "0603B39.BIN"
```

## Firmware Download Retry Mechanism

The firmware download retry mechanism is activated in the following cases:

- The Master tries to download firmware files from a specified external server and the server is not available.
- A follower tries to download from the Master but the Master is too busy and cannot accept additional download requests.

This mechanism configures a timer on the card. For the SMC, the timer is 90 seconds; on the other cards, the timer is 45 seconds. For the first five minutes of card bootup, retries regarding to the time set by this timer. After five minutes, if the firmware file is still not successfully retrieved, the retry period is assigned a value of 3600 seconds (hourly basis), and no syslog message prints.

The retry mechanism prints the following messages to the console or SYSLOG file in the first five minutes.

```
NOV 01 16:31:36 UMS: Info Retry firmware download:  from IP
47.11.215.82 for terminal i2004
File "/ums/i2004.fw" not found or permission problem
NOV 01 16:31:36 UMS: Error FTP(47.11.215.82):  send command
failed
NOV 01 16:31:36 UMS: Info Retry firmware download:  from IP
47.11.215.82 for terminal i2002
File "/ums/i2002.fw" not found or permission problem
NOV 01 16:31:36 UMS: Error FTP(47.11.215.82):  send command
failed
```

At five minutes, if no firmware arrives on the VGMC or Signaling Server, and the Master firmware version is unknown:

```
NOV 01 16:34:49 UMS: 5 Mins passed:  Not all F/W downloaded
successfully.
```

## Disk Space Check

When you download firmware to the VGMC using EM/OTM or the firmwarefileGetI2004 command, before the actual FTP transfer occurs, the disk space on the VGMC is checked. This disk space check ensures that the VGMC has enough space for normal operation after the download. The check sequence is as follows:

1. Check C: drive disk space.
2. Check A: drive disk space (if C has insufficient space).

The calculation is as follows:

1. If the firmware file to download exists in /C:/fw, then check if the /C: has 500 KB (reserve disk space for VGMC application)
   OR
   If the firmware file to download does not exist in /C:/fw, then check if the C: has 1.45 MB (500 (minimum) + 950 (estimate size of compressed fw)).
   If C: disk space is sufficient, proceed with download.
2. Otherwise, check that A: has either 0 KB (firmware file exists on A:) OR 950 KB (new firmware file for the A:).
   If A: disk space is sufficient, proceed with download.
3. Otherwise, the system prints error messages to the console and sends an alarm.

### Firmware file backup on drive

A backup copy of the firmware file is made in the Leader/Master under the following circumstances:

1. When you download firmware to the Signaling Server through OTM/EM, using the firmwarefileGetI2004 command, or through an external server.
2. At VGMC bootup time, when Leader/Master download firmware from the external server (because at runtime, firmware is directly downloaded and compressed to /fw, a backup is not necessary.

The same disk check described in "Disk Space Check" (page 431) is performed for the backup.

For the backup progress, the Signaling Server takes only 2 to 5 seconds. The VGMCs take longer: ITG-P takes about 2 to 3 minutes for each file and the SMC takes about 1 to 2 minutes for each file depending on how busy the card is.

During the download, a power loss, car, reboot, or other event may cause an incomplete backup to the C: drive or A: drive. When the card boots up and the application starts, the application detects and does not use an incomplete or corrupt firmware file.

## IP Softphone 2050 software installation

You must manually install the IP Softphone 2050 soft client application software on the PC. There is no download mechanism from OTM/EM.

# Security features

## New security features in CS 1000 Release 5.x

CS 1000 Release 5.x includes the following security features:

- Secure remote access is provided by Secure Shell (SSH).
- Security for individual call streams is provided by the Media Security feature.
- Security for ELAN subnets is provided by the system management security and ISSS, which is based on the industry standard IP Security (IPsec).
- Security for SIP signaling is provided by Transport Layer Security (TLS).
- Security for data exchanges between the IP Phones and the Signaling Server is provided by Secure UNIStim signaling, which requires a Secure Multimedia Controller (SMC) 2450 device.

## Security for SIP signaling – SIP TLS

### TLS Connection Establishment

[graphic]

### TLS Basic Call Flow

[graphic]

### CLI Commands

#### SIPGwShow

You can use this command to display primary and secondary proxy transport types and TLS usage. The URI scheme appears in the channel table at the end of the output from this command.

```
oam> SIPGwShow
SIPNPM Status :  Active
Primary Proxy IP address :  47.11.232.110
Secondary Proxy IP address :  47.9.194.167
Primary Proxy port :  5061
```

```
Secondary Proxy port :  5061
Primary Proxy Transport :  TLS
Secondary Proxy Transport :  TLS
Active Proxy :  Primary :Registered
Time To Next Registration :  7 Seconds Channels Busy / Idle /
Total :  30 / 34 / 64
Stack version :  4.0.0.30
TLS Security Policy :  Best Effort
SIP Gw Registration Trace :  OFF
Output Type Used :  RPT
Channel tracing :  -1
Channel id should be a non-zero value
```

### SIPCallTrace

You can use this command to show the transport and URI scheme.

### SIPTLSConfigShow

Display TLS configuration parameters of the entire system, including client and server session caching parameters, the certificate for the local system, and the certificates that are configured.

```
pdt> SIPTLSConfigShow
TLS Parameter Details
**********************

TLS Configuration Details
-------------------------------------------
TLS Security Policy:  Best Effort
TLS Client Authentication:  Yes
TLS session caching size:  200
TLS session timeout:  600 (s)
TLS renegotiation threshold:  20000000 (packets)
TLS local certificate:  /u/ssl/cert/0.pem
There are 1 CA certificate(s) configured:
-------------------------------------------

/u/ssl/ca/73bf4006.r0
X509:Subject
</C=CA/ST=Ontario/L=Belleville/O=Nortel/OU=BVW/CN=bvwib
mem.ca.nortel.com>X509:Issuer
</C=CA/ST=Ontario/L=Belleville/O=Nortel/OU=BVW/CN=bvwib
mem.ca.nortel.com>
X509:Public Key 1024 (bits)
X509:Valid From <Feb 27 14:35:37 2007 GMT>
X509:Valid Until <Feb 26 14:35:37 2037 GMT>
Session caching parameters - Client
-------------------------------------------
```

```
NumSessionAllowed 200
NumSessionInCache 23
NumConnectionsReceived 1277
NumConnectionsRenegotiation 2
NumConnectionsFinished 1275
NumConnectionsAccepted 0
NumSessionCacheHit 1238
NumSessionCacheMiss 0
NumSessionTimeout 0
NumSessionCallBackHit 0
NumCacheOverFlow 0

Session caching parameters - Server
-------------------------------------------
NumSessionAllowed 200
NumSessionInCache 2
NumConnectionsReceived 0 NumConnectionsRenegotiation 0
NumConnectionsFinished 0
NumConnectionsAccepted 3
NumSessionCacheHit 0
NumSessionCacheMiss 0
NumSessionTimeout 0
NumSessionCallBackHit 0
NumCacheOverFlow 0
pdt>
```

### SIPTLSSessionShow

Display the details of all SIP TLS sessions or sessions associated with a server IP address. This command shows existing sessions (in connected state and persistent), cached sessions, and the uptime and cipher suites, but does not show key information.

```
pdt> SIPTLSSessionShow

SIPTLSSessionShow

Usage :SIPTLSSessionShow <srvIP/srvIP,port/ALL>
srvIP : display the TLS Session details for given server IP
srvIP, port :  display the TLS Session details for given
server IP and Port
ALL : display all the existing TLS session details

pdt> SIPTLSSessionShow all
RemoteIP Port UpTime SSL SslSession Cipher:  Version Name
Bits
```

```
-------------- --- -------- --------- ----------
---------- --------- ----
Totally there are 0 TLS session caching entries.
```

**SIPMessageTrace**
Configure filtering criteria for message tracing.

The capture buffer is intentionally not freed after the file is saved in case a problem occurs and you need to retrieve the file. However, be careful not to allocate a large buffer and leave it on the card in case later processing on the card requires extra memory.

# SIP NRS on Linux

This section describes troubleshooting the SIP NRS on Linux.

## Linux commands

The following diagnostic commands are helpful in diagnosing application issues:

- traceroute
- free
- ipcs
- ldconfig
- ping

Each command is described on a man page in Linux.

```
spspri -> host name
-p option is to specify the port while performin traceroute.
-s option is the source ip address of the SPS box.

[nortel@spspri ~]$ traceroute -p 5060 spspri
traceroute to spspri.nortel.com (47.11.119.133), 30 hops
max, 38 byte packets
1 spspri (47.11.119.133) 0.089 ms 0.039 ms 0.037 ms
[nortel@spspri ~]$

[nortel@spspri ~]$ traceroute -s 47.11.119.133 spspri
traceroute to spspri.nortel.com (47.11.119.133) from
47.11.119.133, 30 hops max, 38 byte packets
1 spspri (47.11.119.133) 0.086 ms 0.038 ms 0.036 ms
[nortel@spspri ~]$
[nortel@spspri ~]$

[nortel@spspri ~]$ free
total used free shared buffers cached
Mem:  2074952 1546688 528264 0 133096 710448
```

```
-/+ buffers/cache:  703144 1371808
Swap:  4192912 160 4192752
[nortel@spspri ~]$
[nortel@spspri ~]$

[nortel@spspri ~]$ ipcs

------ Shared Memory Segments --------
key shmid owner perms bytes nattch status
0x6107c07f 1343488 nortel 770 2864 13
0x6207c07f 1376257 nortel 770 2097152 8
0x6307c07f 1409026 nortel 770 1048576 9
0x6407c07f 1441795 nortel 770 1048576 9
0x6507c07f 1474564 nortel 770 1048576 8
0x6607c07f 1507333 nortel 770 2097152 9
0x6707c07f 1540102 nortel 770 2097152 9
0x6807c07f 1572871 nortel 770 4194304 13
0x6907c07f 1605640 nortel 770 1048576 8
0x6a07c07f 1638409 nortel 770 524288 8
0x6b07c07f 1671178 nortel 770 2097152 9
0x6c07c07f 1703947 nortel 770 2097152 9
0x6d07c07f 1736716 nortel 770 1048576 8
0x7807c0a7 1310733 nortel 666 4096 5


------ Semaphore Arrays --------
key semid owner perms nsems
0x61078001 1310721 nortel 770 1
0x62078001 1343490 nortel 770 1
0x63078001 1376259 nortel 770 1
0x64078001 1409028 nortel 770 1
0x65078001 1441797 nortel 770 1
0x66078001 1474566 nortel 770 1
0x67078001 1507335 nortel 770 1
0x68078001 1540104 nortel 770 1
0x69078001 1572873 nortel 770 1
0x6a078001 1605642 nortel 770 1
0x6b078001 1638411 nortel 770 1
0x6c078001 1671180 nortel 770 1
0x6d078001 1703949 nortel 770 1

------ Message Queues --------
key msqid owner perms used-bytes messages
0x6107c080 1212417 nortel 770 0 0
0x6207c080 1245186 nortel 770 0 0
0x6307c080 1277955 nortel 770 0 0
0x6407c080 1310724 nortel 770 0 0
```

```
0x6507c080 1343493 nortel 770 0 0
0x6607c080 1376262 nortel 770 0 0
0x6707c080 1409031 nortel 770 0 0
0x6807c080 1441800 nortel 770 0 0
0x6907c080 1474569 nortel 770 0 0
0x6a07c080 1507338 nortel 770 0 0
0x6b07c080 1540107 nortel 770 0 0
0x6c07c080 1572876 nortel 770 0 0

[nortel@spspri ~]$
[nortel@spspri ~]$
[nortel@spspri ~]$ /sbin/ldconfig
-v /sbin/ldconfig:  Path '/opt/nortel/Snmp-Daemon-TrapLib
/lib' given more than once
/sbin/ldconfig:  Path '/opt/nortel/open/osip' given more
than once
/usr/X11R6/lib:
libX11.so.6 -> libX11.so.6.2
libGLw.so.1 -> libGLw.so.1.0
libXevie.so.1 -> libXevie.so.1.0
libXcursor.so.1 -> libXcursor.so.1.0.2
libXrandr.so.2 -> libXrandr.so.2.0
libXext.so.6 -> libXext.so.6.4
libXfixes.so.3 -> libXfixes.so.3.0
libXtst.so.6 -> libXtst.so.6.1
libXinerama.so.1 -> libXinerama.so.1.0
libXaw3d.so.7 -> libXaw3d.so.7.0
libfontenc.so.1 -> libfontenc.so.1.0
libxkbfile.so.1 -> libxkbfile.so.1.0
libdpstk.so.1 -> libdpstk.so.1.0
libXxf86vm.so.1 -> libXxf86vm.so.1.0
libOSMesa.so.4 -> libOSMesa.so.4.0
libXaw.so.6 -> libXaw.so.6.1
libXfont.so.1 -> libXfont.so.1.5
libXaw.so.7 -> libXaw.so.7.0
libXv.so.1 -> libXv.so.1.0
libXvMC.so.1 -> libXvMC.so.1.0
libXmuu.so.1 -> libXmuu.so.1.0
libXxf86misc.so.1 -> libXxf86misc.so.1.1
libXxf86rush.so.1 -> libXxf86rush.so.1.0
libICE.so.6 -> libICE.so.6.3
libXxf86dga.so.1 -> libXxf86dga.so.1.0
libXt.so.6 -> libXt.so.6.0
libXrender.so.1 -> libXrender.so.1.2.2
libXmu.so.6 -> libXmu.so.6.2
libdps.so.1 -> libdps.so.1.0
libXcomposite.so.1 -> libXcomposite.so.1.0
```

```
libxkbui.so.1 -> libxkbui.so.1.0
libXft.so.1 -> libXft.so.1.1
libGL.so.1 -> libGL.so.1.2
libSM.so.6 -> libSM.so.6.0
libXpm.so.4 -> libXpm.so.4.11
libXRes.so.1 -> libXRes.so.1.0
libXss.so.1 -> libXss.so.1.0
libFS.so.6 -> libFS.so.6.0
libXi.so.6 -> libXi.so.6.0
libXdamage.so.1 -> libXdamage.so.1.0
libI810XvMC.so.1 -> libI810XvMC.so.1.0
libpsres.so.1 -> libpsres.so.1.0
libXft.so.2 -> libXft.so.2.1.2
libXTrap.so.6 -> libXTrap.so.6.4 /opt/nortel/base/open/op
enssl/openssl-0.9.7l/linux-i686/lib:
libssl.so.0.9.7 -> libssl.so.0.9.7
libcrypto.so.0.9.7 -> libcrypto.so.0.9.7 /opt/nortel/open
/osip:
libosipparser2.so.3 -> libosipparser2.so.3.0.0
libosip2.so.3 -> libosip2.so.3.0.0
/opt/nortel/Snmp-Daemon-TrapLib/lib:
libSnmpTrapHandler.so -> libSnmpTrapHandler.so
/opt/nortel/base/patchutil/binary:
libee_testexeclib.1.0.1.so -> libee_testexeclib.1.0.1.so
libee_testlib.1.0.3.so -> libee_testlib.1.0.3.so
libee_infrastructure.1.0.0.so -> libee_infrastructure.1.
0.0.so
libslgapi.so -> libslgapi.so
libee_testlib.so -> libee_testlib.so
libOamFm.so -> libOamFm.so
libee_patcher_tap.so -> libee_patcher_tap.so
libee_patcher_tap.1.0.0.so -> libee_patcher_tap.1.0.0.so
libee_patcher.1.0.0.so -> libee_patcher.1.0.0.so
libee_testlib.1.0.1.so -> libee_testlib.1.0.1.so
libee_patcher_loader.so -> libee_patcher_loader.so
libpatch_event.so -> libpatch_event.so
libee_interceptor.1.0.0.so -> libee_interceptor.1.0.0.so
libee_interceptor.so -> libee_interceptor.so
libtap_slr_a.so -> libtap_slr_a.so
libmercapi.so -> libmercapi.so
libee_patcher_loader.1.0.0.so -> libee_patcher_loader.1.
0.0.so
libee_fault.1.0.0.so -> libee_fault.1.0.0.so
libee_patcher.so -> libee_patcher.so
libee_fault.so -> libee_fault.so
libee_infrastructure.so -> libee_infrastructure.so
libhwServices.so -> libhwServices.so
```

```
libee_testexeclib.so -> libee_testexeclib.so
libee_testlib.1.0.2.so -> libee_testlib.1.0.2.so
/lib:
libdevmapper.so.1.01 -> libdevmapper.so.1.01
libpam_misc.so.0 -> libpam_misc.so.0.77
libSegFault.so -> libSegFault.so
libanl.so.1 -> libanl-2.3.4.so
libsepol.so.1 -> libsepol.so.1
libNoVersion.so.1 -> libNoVersion-2.3.4.so
libthread_db.so.1 -> libthread_db-1.0.so
libselinux.so.1 -> libselinux.so.1
libm.so.6 -> libm-2.3.4.so
libdb-3.3.so -> libdb-3.3.so
libgcc_s.so.1 -> libgcc_s-3.4.5-20051201.so.1
libpcre.so.0 -> libpcre.so.0.0.1
libcidn.so.1 -> libcidn-2.3.4.so
libcom_err.so.2 -> libcom_err.so.2.1
libpamc.so.0 -> libpamc.so.0.77
libnss_compat.so.1 -> libnss1_compat.so.1
libnss_nisplus.so.2 -> libnss_nisplus-2.3.4.so
libe2p.so.2 -> libe2p.so.2.3
libacl.so.1 -> libacl.so.1.1.0
libcrypt.so.1 -> libcrypt-2.3.4.so
libnss_dns.so.1 -> libnss1_dns.so.1
libnsl.so.1 -> libnsl-2.3.4.so
libcap.so.1 -> libcap.so.1.10
libblkid.so.1 -> libblkid.so.1.0
libnss_compat.so.2 -> libnss_compat-2.3.4.so
libdevmapper.so.1.00 -> libdevmapper.so.1.00
libaudit.so.0 -> libaudit.so.0.0.0
libdb.so.3 -> libdb2.so.3
libproc-3.2.3.so -> libproc-3.2.3.so
libnss_files.so.2 -> libnss_files-2.3.4.so
libnss_ldap.so.2 -> libnss_ldap-2.3.4.so
libasound.so.2 -> libasound.so.2.0.0
libBrokenLocale.so.1 -> libBrokenLocale-2.3.4.so
libnss_nis.so.1 -> libnss1_nis.so.1
libdl.so.2 -> libdl-2.3.4.so
libdb-4.1.so -> libdb-4.1.so
libnss_files.so.1 -> libnss1_files.so.1
libattr.so.1 -> libattr.so.1.1.0
ld-linux.so.2 -> ld-2.3.4.so
libss.so.2 -> libss.so.2.0
libssl.so.4 -> libssl.so.0.9.7a
librt.so.1 -> librt-2.3.4.so
libpthread.so.0 -> libpthread-0.10.so
libdevmapper.so.1.02 -> libdevmapper.so.1.02
```

```
libc.so.6 -> libc-2.3.4.so
libuuid.so.1 -> libuuid.so.1.2
libresolv.so.2 -> libresolv-2.3.4.so
libpam.so.0 -> libpam.so.0.77
libnss_dns.so.2 -> libnss_dns-2.3.4.so
libnss_nis.so.2 -> libnss_nis-2.3.4.so
libext2fs.so.2 -> libext2fs.so.2.4
libtermcap.so.2 -> libtermcap.so.2.0.8
libnss_hesiod.so.2 -> libnss_hesiod-2.3.4.so
libcrypto.so.4 -> libcrypto.so.0.9.7a
libdb-4.2.so -> libdb-4.2.so
libutil.so.1 -> libutil-2.3.4.so
/usr/lib:
libncursesw.so.5 -> libncursesw.so.5.4
libgnome-2.so.0 -> libgnome-2.so.0.800.0
libsasl.so.7 -> libsasl.so.7.1.11
libgcrypt.so.11 -> libgcrypt.so.11.1.1
libslang-utf8.so.1 -> libslang-utf8.so.1.4.9
libstunnel.so -> libstunnel.so
libstdc++-libc6.2-2.so.3 -> libstdc++-3-libc6.2-2-2.10.0
.so
libgamin-1.so.0 -> libgamin-1.so.0.1.1
libstdc++.so.2.7.2 -> libstdc++.so.2.7.2.8
libfam.so.0 -> libfam.so.0.0.0
libkrbafs.so.0 -> libkrbafs.so.0.0.0
libradius-1.0.1.so -> libradius.so
libgettextlib-0.14.1.so -> libgettextlib.so
libxslt.so.1 -> libxslt.so.1.1.11
libgettextsrc-0.14.1.so -> libgettextsrc.so
libgssrpc.so.3 -> libgssrpc.so.3.0
libelf.so.1 -> libelf-0.97.so
libglib-2.0.so.0 -> libglib-2.0.so.0.400.7
libnetsnmp.so.10 -> libnetsnmp.so.10.0.1
libethereal.so.0 -> libethereal.so.0.0.1
libORBit-2.so.0 -> libORBit-2.so.0.0.0
libldap_r-2.2.so.7 -> libldap_r-2.2.so.7.0.6
libexslt.so.0 -> libexslt.so.0.8.9
libnss_nis.so.1 -> libnss1_nis.so
libwrap.so.0 -> libwrap.so.0.7.6
libreadline.so.4 -> libreadline.so.4.3
libsensors.so.3 -> libsensors.so.3.0.5
libhal-storage.so.0 -> libhal-storage.so.0.0.0
libg++.so.2.7.2 -> libg++.so.2.7.2.8
librpmbuild-4.3.so -> librpmbuild-4.3.so
libpcap.so.0.8.3 -> libpcap.so.0.8.3
libdbus-glib-1.so.0 -> libdbus-glib-1.so.0.0.0
libstdc++.so.2.9 -> libstdc++.so.2.9.dummy
```

```
libtiff.so.3 -> libtiff.so.3.6
libfontconfig.so.1 -> libfontconfig.so.1.0.4
libcmaX.so.1 -> libcmaX.so.1.0
libpython2.3.so.1.0 -> libpython2.3.so.1.0
libwnck-1.so.4 -> libwnck-1.so.4.9.0
libttf.so.2 -> libttf.so.2.3.0
liblwres.so.1 -> liblwres.so.1.1.2
libsasl2.so.2 -> libsasl2.so.2.0.19
libmenuw.so.5 -> libmenuw.so.5.4
libgthread-1.2.so.0 -> libgthread-1.2.so.0.0.10
lib-org-xml-sax.so.5 -> lib-org-xml-sax.so.5.0.0
libgcj.so.5 -> libgcj.so.5.0.0
libgpg-error.so.0 -> libgpg-error.so.0.1.3
libexpect5.42.so -> libexpect5.42.so
libisccc.so.0 -> libisccc.so.0.1.0
libformw.so.5 -> libformw.so.5.4
libopcodes-2.15.92.0.2.so -> libopcodes-2.15.92.0.2.so
libpanelw.so.5 -> libpanelw.so.5.4
libtcl8.4.so -> libtcl8.4.so
libbz2.so.1 -> libbz2.so.1.0.2
libnss_dns.so.1 -> libnss1_dns.so
libnetsnmphelpers.so.10 -> libnetsnmphelpers.so.10.0.1
libwiretap.so.0 -> libwiretap.so.0.0.1
libhpev.so.1 -> libhpev.so.1.0
liblber-2.2.so.7 -> liblber-2.2.so.7.0.6
libpopt.so.0 -> libpopt.so.0.0.0
libutempter.so.0 -> libutempter.so.0.5.5
libhistory.so.4 -> libhistory.so.4.3
libgdk-x11-2.0.so.0 -> libgdk-x11-2.0.so.0.400.13
libbonoboui-2.so.0 -> libbonoboui-2.so.0.0.0
libdb.so.2 -> libdb1.so.2
libmp.so.3 -> libmp.so.3.1.7
libgmp.so.3 -> libgmp.so.3.3.3
librpmdb-4.3.so -> librpmdb-4.3.so
libfreetype.so.6 -> libfreetype.so.6.3.7
librpmio-4.3.so -> librpmio-4.3.so
libk5crypto.so.3 -> libk5crypto.so.3.0
libstdc++.so.6 -> libstdc++.so.6.0.3
libnetsnmphelpers.so.5 -> libnetsnmphelpers.so.5.1.2
libkadm5clnt.so.5 -> libkadm5clnt.so.5.1
libjpeg.so.62 -> libjpeg.so.62.0.0
libgnomecanvas-2.so.0 -> libgnomecanvas-2.so.0.800.0
libgtk-x11-2.0.so.0 -> libgtk-x11-2.0.so.0.400.13
libgdk_pixbuf-2.0.so.0 -> libgdk_pixbuf-2.0.so.0.400.13
libesd.so.0 -> libesd.so.0.2.35
libgdbm.so.2 -> libgdbm.so.2.0.0
libkadm5srv.so.5 -> libkadm5srv.so.5.1
```

```
libbonobo-activation.so.4 -> libbonobo-activation.so.4.0
.0
libgnomeui-2.so.0 -> libgnomeui-2.so.0.800.0
libstdc++.so.2.8 -> libstdc++.so.2.8.0
libmagic.so.1 -> libmagic.so.1.0
libgconf-2.so.4 -> libgconf-2.so.4.1.0
libstartup-notification-1.so.0 -> libstartup-notificatio
n-1.so.0.0.0
lib-org-w3c-dom.so.5 -> lib-org-w3c-dom.so.5.0.0
libORBitCosNaming-2.so.0 -> libORBitCosNaming-2.so.0.0.0
libhesiod.so.0 -> libhesiod.so.0
libgmodule-2.0.so.0 -> libgmodule-2.0.so.0.400.7
libgnomevfs-2.so.0 -> libgnomevfs-2.so.0.800.2
libgssapi_krb5.so.2 -> libgssapi_krb5.so.2.2
libnetsnmp.so.5 -> libnetsnmp.so.5.1.2
libncurses.so.5 -> libncurses.so.5.4
libpango-1.0.so.0 -> libpango-1.0.so.0.600.0
libIDL-2.so.0 -> libIDL-2.so.0.0.0
libnetsnmpmibs.so.10 -> libnetsnmpmibs.so.10.0.1
libcpqci.so.1 -> libcpqci.so.1.0
libdns.so.16 -> libdns.so.16.0.0
libart_lgpl_2.so.2 -> libart_lgpl_2.so.2.3.16
libeap-1.0.1.so -> libeap.so
libkrb5.so.3 -> libkrb5.so.3.2
libnss_files.so.1 -> libnss1_files.so
libbonobo-2.so.0 -> libbonobo-2.so.0.0.0
libgmpxx.so.3 -> libgmpxx.so.3.0.5
libxml2.so.2 -> libxml2.so.2.6.16
libgmodule-1.2.so.0 -> libgmodule-1.2.so.0.0.10
libbeecrypt.so.6 -> libbeecrypt.so.6.2.0
libpangoxft-1.0.so.0 -> libpangoxft-1.0.so.0.600.0
libpangox-1.0.so.0 -> libpangox-1.0.so.0.600.0
libstdc++-libc6.1-1.so.2 -> libstdc++-2-libc6.1-1-2.9.0.
so
liblockdev.so.1 -> liblockdev.so.1.0.1
libungif.so.4 -> libungif.so.4.1.3
libdbus-1.so.0 -> libdbus-1.so.0.0.0
libstdc++.so.5 -> libstdc++.so.5.0.7
libkrb4.so.2 -> libkrb4.so.2.0
libglib-1.2.so.0 -> libglib-1.2.so.0.0.10
libatk-1.0.so.0 -> libatk-1.0.so.0.800.0
libnetsnmptrapd.so.5 -> libnetsnmptrapd.so.5.1.2
libbfd-2.15.92.0.2.so -> libbfd-2.15.92.0.2.so
libnetsnmpagent.so.5 -> libnetsnmpagent.so.5.1.2
libisccfg.so.0 -> libisccfg.so.0.0.11
libnetsnmpmibs.so.5 -> libnetsnmpmibs.so.5.1.2
libnetsnmpagent.so.10 -> libnetsnmpagent.so.10.0.1
```

```
libltdl.so.3 -> libltdl.so.3.1.0
libpcreposix.so.0 -> libpcreposix.so.0.0.0
libaudiofile.so.0 -> libaudiofile.so.0.0.2
libnetsnmptrapd.so.10 -> libnetsnmptrapd.so.10.0.1
libhal.so.0 -> libhal.so.0.0.0
libnewt.so.0.51 -> libnewt.so.0.51.6
lib-gnu-java-awt-peer-gtk.so.5 -> lib-gnu-java-awt-peer-
gtk.so.5.0.0
libcmacommon.so.1 -> libcmacommon.so.1.0
libgpm.so.1 -> libgpm.so.1.19.0
libdes425.so.3 -> libdes425.so.3.0
libform.so.5 -> libform.so.5.4
libgobject-2.0.so.0 -> libgobject-2.0.so.0.400.7
libcrack.so.2 -> libcrack.so.2.7
libldap-2.2.so.7 -> libldap-2.2.so.7.0.6
libz.so.1 -> libz.so.1.2.1.2
libuser.so.1 -> libuser.so.1.1.1
libgtop-2.0.so.4 -> libgtop-2.0.so.4.0.0
librpm-4.3.so -> librpm-4.3.so
libesddsp.so.0 -> libesddsp.so.0.2.35
libpng12.so.0 -> libpng12.so.0.1.2.7
libgthread-2.0.so.0 -> libgthread-2.0.so.0.400.7
libmenu.so.5 -> libmenu.so.5.4
libglade-2.0.so.0 -> libglade-2.0.so.0.0.4
libdb_cxx-4.2.so -> libdb_cxx-4.2.so
libORBit-imodule-2.so.0 -> libORBit-imodule-2.so.0.0.0
libgdk_pixbuf_xlib-2.0.so.0 -> libgdk_pixbuf_xlib-2.0.so
.0.400.13
libpanel.so.5 -> libpanel.so.5.4
libsnmp.so.5 -> libsnmp.so.5.1.2
libkdb5.so.4 -> libkdb5.so.4.0
libparted-1.6.so.12 -> libparted.so
libgnome-keyring.so.0 -> libgnome-keyring.so.0.0.1
libexpat.so.0 -> libexpat.so.0.5.0
libcmapeer.so.1 -> libcmapeer.so.1.0
libisc.so.7 -> libisc.so.7.1.5
libhpasmintrfc.so.1 -> libhpasmintrfc.so.1.0
libpangoft2-1.0.so.0 -> libpangoft2-1.0.so.0.600.0
libobjc.so.1 -> libobjc.so.1.0.0
libnss_compat.so.1 -> libnss1_compat.so
/usr/X11R6/lib/tls:  (hwcap:  0x8000000000000000)
/lib/tls:  (hwcap:  0x8000000000000000)
/usr/lib/tls:  (hwcap:  0x8000000000000000)
/usr/lib/sse2:  (hwcap:  0x4000000)
libmp.so.3 -> libmp.so.3.1.7
libgmp.so.3 -> libgmp.so.3.3.3
libgmpxx.so.3 -> libgmpxx.so.3.0.5
```

```
/lib/tls/i686: (hwcap: 0x8008000000000000)
libthread_db.so.1 -> libthread_db-1.0.so
libm.so.6 -> libm-2.3.4.so
librt.so.1 -> librt-2.3.4.so
libc.so.6 -> libc-2.3.4.so
libpthread.so.0 -> libpthread-2.3.4.so
libdb-4.2.so -> libdb-4.2.so
/lib/tls/i586: (hwcap: 0x8004000000000000)
libthread_db.so.1 -> libthread_db-1.0.so
libm.so.6 -> libm-2.3.4.so
librt.so.1 -> librt-2.3.4.so
libc.so.6 -> libc-2.3.4.so
libpthread.so.0 -> libpthread-2.3.4.so
libdb-4.2.so -> libdb-4.2.so
/lib/tls/i486: (hwcap: 0x8002000000000000)
libthread_db.so.1 -> libthread_db-1.0.so
libm.so.6 -> libm-2.3.4.so
librt.so.1 -> librt-2.3.4.so
libc.so.6 -> libc-2.3.4.so
libpthread.so.0 -> libpthread-2.3.4.so
libdb-4.2.so -> libdb-4.2.so
/usr/lib/tls/i686: (hwcap: 0x8008000000000000)
libdb_cxx-4.2.so -> libdb_cxx-4.2.so
/usr/lib/tls/i586: (hwcap: 0x8004000000000000)
libdb_cxx-4.2.so -> libdb_cxx-4.2.so
/usr/lib/tls/i486: (hwcap: 0x8002000000000000)
libdb_cxx-4.2.so -> libdb_cxx-4.2.so
/sbin/ldconfig: Can't create temporary cache file
/etc/ld.so.cache~: Permission denied
[nortel@spspri ~]$
[nortel@spspri ~]$
```

After you install a new shared library, this command updates the shared library symbolic links in /lib.

```
[nortel@spspri ~]$ /sbin/ldconfig -n /lib
[nortel@spspri ~]$
```

## SIP NRS command

Syntax:

**`spcmd - [H|L|O|R|S|V] - [s|t|u|v <defined value>] ...`**

The spcmd command generates a SIP log file at the location /var/opt/nortel/sps/LOG/SIPLogFile.

**Table 115**
**Command description**

| Family | Parameter | Description |
|---|---|---|
| -H | None | Display a help message that describes the proper syntax for this command. |
| -L | -v debug \| info \| all<br><br>-s on \| off | Write debug, information, or all (that is, both) logs in addition to the SIP log file.<br><br>Turn on/off the log types listed by the -v parameter. If no parameter is given, the default is assigned a value of on. Default is on. |
| -O | -v 400 \| 401 \| 407 \| hw \| ss | Display OM report for 400, 401, 407, or 3XX responses as well as the high water (hw) mark for internal queue memory usage and the number of SIP sessions (ss) that have been established. |
| -R | -s force \| wait \| now<br><br>-t now \| 1..99<br><br>-u min \| sec | Execute a shutdown and restart of the application immediately or in some given time unit (min\sec) whether call are executing or not by either forcing the application or waiting for call processing to stop. |
| -S | -s force \| wait<br><br>-t now \| 1..99<br><br>-u min \| sec | Execute a switching of activity from the running application processing to stop where a timer value can be given. |

**Table 115**
**Command description (cont'd.)**

| Family | Parameter | Description |
|---|---|---|
| - V | -v app \| stack \| all | Show the version of the application, oSIP stack, or both. |
| - D | -v backup <file>,<file><br><br>-v restore <file>,<file>,<file><br><br>-v gknrsdataconvert <parm1-parm5><br><br>-v siproutingtest <parm1-parm7> | Database base command access. Select the base command to run along with the parameters that are expected for the command. Results are written to the report file specified. |

```
[nortel@spspri ~]$ cd /opt/nortel/sps/bin/
```

The following example shows how to log only DEBUG messages in the SIP Log file.

```
[nortel@spspri bin]$ ./spcmd -L -v debug
```

The following example shows how to log only INFO messages in the SIP log file.

```
nortel@spspri bin]$ ./spcmd -L -v info
[nortel@spspri bin]$
```

The following example shows how to log all messages in the SIP log file.

```
[nortel@spspri bin]$ ./spcmd -L -v all
[nortel@spspri bin]$
```

The following example shows how to enable debugging.

```
[nortel@spspri bin]$ ./spcmd -L -s on
[nortel@spspri bin]$
```

The following example shows how to disable debugging.

```
[nortel@spspri bin]$ ./spcmd -L -s off
[nortel@spspri bin]$
[nortel@spspri bin]$
[nortel@spspri bin]$
```

> ***Note:*** Currently the debugging messages are always turned on and cannot be turned off.

The following example shows how to gracefully disable SPS. In this case, SPS proxy is running but out of service. This command returns 0 when successful.

```
[nortel@spspri bin]$ ./spcmd -R -s force 0
[nortel@spspri bin]$
[nortel@spspri bin]$
```

The following example shows how to gracefully enable SPS, so that SPS is in service. This command returns 0 when successful.

```
[nortel@spspri bin]$ ./spcmd -S -s force0
[nortel@spspri bin]$
```

The following example shows how to shut down the NCS application. This command returns 0 when successful.

```
[nortel@spspri bin]$ ./spcmd -R -c ncs
Stopping the NCS components
Initiating shutdown of "NCS"
ncsd pid(s):  26344 26345 26346
Initiating cleanup:  "IPC resources"
ipcrm -m 1310733
[nortel@spspri bin]$
[nortel@spspri bin]$
```

The following example shows how to start the NCS application. This command returns 0 when successful.

```
[nortel@spspri bin]$ ./spcmd -S -c ncs
Starting the NCS components using uid:500 gid:500 for
startup.
starting:  ncsd
[nortel@spspri bin]$ Starting NCS application.
Restart string:  /opt/nortel/ncs/scripts/ncs-exec.pl
restart
NCS Configuration Role=0
IPPrimary=47.11.119.133 port=16500
```

```
IPAlternate=47.11.119.165 port=16500
Success creating pid file 11614
[nortel@spspri bin]$
```

The following example shows how to shut down the SPS application. In this case, the SPS proxy is unavailable and SPS is out of service. This command returns 0 when successful.

```
[nortel@spspri bin]$ ./spcmd -R -c sps
Stopping the SPS Monitoring using uid:500 gid:500 for SPS
monitor startup.
Killing spsmon pid :  26770
Stopping the SPS components
Initiating shutdown of "SPS"
sipprxd pid(s):  26714 26715 26717 26719 26725 26726 26727
26731
sptlsbroker pid(s):  26757
sptcp pid(s):  26741 26743
spudp pid(s):  26759 26761
Initiating cleanup:  "IPC resources"
ipcrm -m 1343488 -m 1572871 -m 1507333 -m 1540102 -m 1409026
-m 1441795 -m 1703947 -m 1671178 -m 1605640 -m 1474564 -m
1376257 -m 1638409 -m 1736716 -s 1310721 -s 1343490 -s
1441797 -s 1474566 -s 1376259 -s 1409028 -s 1638411 -s
1671180 -s 1507335 -s 1540104 -s 1572873 -s 1605642 -s
1703949 -q 1212417 -q 1245186 -q 1277955 -q 1409031 -q
1441800 -q 1507338 -q 1540107 -q 1310724 -q 1343493 -q
1376262 -q 1474569 -q 1572876
The stop operation completed successfully.
[nortel@spspri bin]$
[nortel@spspri bin]$
```

The following example shows how to start the SPS application. In this case, the SPS proxy restarts and SPS is in service. This command returns 0 when successful. .

```
[nortel@spspri bin]$ ./spcmd -S -c sps
Starting the SPS components using uid:500 gid:500 for
startup.
starting:  sipprxd sptcpcd sptcpsd sptlsbrokerd spudpcd
spudpsd
starting init code.  zhiguoc CS1000_5.0_SPS_RPM4 Monday
June 04 2007 13:10:27 EDT zhiguoc
restart string:  /opt/nortel/sps/scripts/sps-exec.pl
restart
Setting up Semaphores
Proxy (NODE_NAME) nodeName=secondary_sps realm=realmName
Proxy (SW_VERSION) serverType=2 Role=0 DB access=0 0 0
PAssertedIdentitySIP=unknown
PAssertedIdentityTEL=000-000
tls values 2048000 600 2048000 600 1 1
Number of interfaces:  3
I/F udp:0 primary 47.11.119.133:5060 secondary
47.11.119.165:5060
I/F tcp:1 primary 47.11.119.133:5060 secondary
47.11.119.165:5060
I/F tls:2 primary 47.11.119.133:5061 secondary
47.11.119.165:5061
Starting Registrar:  11809
Starting message handler:  11822
Starting Maintenance I/F: 11813
Starting Location:  11812
Building certificate chain for the TLS broker.
Doing ../certs/.
9fcdec2a.r0 => 9fcdec2a.0

Server certificate file not found ...  using default
server.pem.
Make sure certificates are configured correctly before
attempting TLS connections.

Starting router:  11816
Starting UDP Client:  11853
All daemons are running.
Couldn't build certificate chain
Starting the Monitor
The start operation completed successfully.
[nortel@spspri bin]$ using uid:500 gid:500 for SPS monitor
startup.
SPS Mon Has Started (/var/run/sps/spsmon.ipid = 11864)
Starting TLS server:  11851 Starting TLS client:  11851
Starting UDP Server:  11855 Starting TCP Client:  11836
[nortel@spspri bin]$
[nortel@spspri bin]$
[nortel@spspri bin]$
Starting TCP Server:  11838
[nortel@spspri bin]$
```

The following example shows how to display the version of the SPS application.

```
[nortel@spspri bin]$ ./spcmd -V -v app

Network Routing Service (5.00.31.01) SPS on Linux.
[nortel@spspri bin]$
[nortel@spspri bin]$
```

The following example shows how to display the version of the oSIP stack.

```
[nortel@spspri bin]$ ./spcmd -V -v stack

SIP stack (02.02.08) oSIP2 on Linux
[nortel@spspri bin]$
[nortel@spspri bin]$
```

The following example shows how to display the version of the SPS application and oSIP stack.

```
[nortel@spspri bin]$ ./spcmd -V -v all

Network Routing Service (5.00.31.01) SPS on Linux.
SIP stack (02.02.08) oSIP2 on Linux
[nortel@spspri bin]$
[nortel@spspri bin]$
```

The following example shows how to swap the active and standby pointers. If the database is changed the command returns 0. If the command is unsuccessful, it returns 1047.

```
[nortel@spspri bin]$ ./spcmd -D -d cutover
0[nortel@spspri bin]$
```

The following example shows how to swap back the active and standby pointers, if the cutover command was successful. If the database is changed the command returns 0. If the command is unsuccessful, it returns 1048.

```
[nortel@spspri bin]$ ./spcmd -D -d revert
0[nortel@spspri bin]$
```

The following example shows how to copy the table from standby to active schema. This command returns 0 only if the cutover command is executed. If this command is unsuccessful, it returns 1048.

```
[nortel@spspri bin]$ ./spcmd -D -d commit
1048[nortel@spspri bin]$
[nortel@spspri bin]$
```

The following example shows how to copy the table from standby to active schema. This command returns 0 only if the database is changed or the cutover command is run after the database is changed. If this command is unsuccessful, it returns 1048.

```
[nortel@spspri bin]$ ./spcmd -D -d rollback
0 [nortel@spspri bin]$
```

The following example shows how to execute cutover and then commit. This command returns 0 if the database is changed. If this command is unsuccessful, it returns 1047.

```
[nortel@spspri bin]$ ./spcmd -D -d commitnow 1047 [nortel@s
pspri bin]$ [nortel@spspri bin]$
```

The following example shows how to back up the database. This command stores the nrsback.tar file in /var/opt/nortel/sps/backup/tar, and nrsback.bak, which is the previous nrsback.tar. Returns 0 if successful. The result is stored in an XML file in /var/opt/nortel/sps/LOG/backup.xml..

```
[nortel@spspri bin]$ ./spcmd -D -d backup
0 [nortel@spspri bin]$
[nortel@spspri bin]$
```

The following example shows how to restore the database from /var/opt/nortel/sps/backup/tar/nrsback.tar. The result is stored in an XML file in /var/opt/nortel/sps/LOG/restoresum.xml.

```
[nortel@spspri bin]$ ./spcmd -D -d restore
0 [nortel@spspri bin]$
[nortel@spspri bin]$
[nortel@spspri bin]$
```

The following example shows how to use the route test tool. Use the following parameters in this command: –m = Service domain, -p = phone context (the L1Domain name, if the DN type is UDP; "L0Domain name.L1Domain name" if DN type is CDP; or null if DN type is E.164 International), -i = originating endpoint IP address, -n = DN to query, -w = schema.

The result is stored in an XML file at /var/opt/nortel/sps/LOG/siptest.xml..

```
[nortel@spspri bin]$ ./spcmd -D -d siptest -m nortel.com -p
udp -i 47.11.30.83 - n 2000 -w active
0 [nortel@spspri bin]$
```

The following example shows how to perform SIP tests using special route test tools to test the route on the second collaborative server SPS. The results are stored in an XML file at /var/opt/nortel/sps/LOG/.

```
[nortel@spspri bin]$ ./spcmd -D -d sipnametest -m
nortel.com -p cdp.udp -i 47.11.30.83 -w active
0 [nortel@spspri bin]$

[nortel@spspri bin]$ ./spcmd -D -d sipzonetest -m
nortel.com -p cdp.udp -w active
0 [nortel@spspri bin]$

[nortel@spspri bin]$ ./spcmd -D -d sipeptest -m nortel.com
-p cdp.udp -n 2000 -w active
0 [nortel@spspri bin]$
```

The following example shows how to convert the gk 3.0 database to the SPS database. This command copies the gkbackup.tar at /var/opt/nortel/sps/backup/tar. If the gkbackup.tar file does not exist, then the command returns 1078. During the conversion, the nrsback.tar file is also required in /var/opt/nortel/sps/backup/tar.

```
[nortel@spspri bin]$ ./spcmd -D -d gkcvt -m nortel.com -l
udp
1078 [nortel@spspri bin]$
```

# Linux Base

## Troubleshooting

### Changes made to /etc/hosts overwritten

Changes to the /etc/hosts by logging on as root and editing the file are not backed up and restored during an update or reinstallation.

Currently, the LinuxBase does not support the addition of explicit hosts, because it provides a way to specify up to three DNS servers (which handle host resolving).

Specify the IP address directly, rather than using a host name.

### Adding a host to /etc/hosts

Add a host to /etc/hosts in the event that:

- The DNS servers are not provisioned on the network.
- The DNS servers are unresponsive (server is down).
- The required host is not specified on the DNS servers.
- The IP address cannot be explicitly supplied.

Added the host to /etc/hosts by doing the following:

1. Log on to the server as root.
2. Run the following line:
   ```
   echo [IP Address] [Hostname.Domain] [Hostname] >>
   [File]
   (ex:  echo 127.0.0.1 testhost.ca.nortel.com testhost >>
   /etc/hosts)
   ```

   ***Note:*** This process should exclusively be used by Nortel personnel. Changes made using this process cannot be backed up or recovered and cannot survive an upgrade or reinstallation.

## LinuxBase command reference

### appinstall

Syntax:

**`appinstall --test <dir>`**

Install Nortel applications. You are prompted to provide root password to execute this command.

Use the --test option only for testing. Use the <dir> option to install applications from a specified directory instead of CD.

### appstart

Syntax:

**`appstart [start, stop, restart, status] <app_name command>`**

Start, stop, or restart Nortel applications. You are prompted to provide the root password to execute this command.

### appVersionShow

Syntax:

**`appVersionShow`**

Print the server application software version.

```
[nortel@hp2 ~]$ appVersionShow
APP_INSTALLED=CS1000 Element Manager
APP_VERSION=4.90-08
sunAm 4.90-08
Jboss-Quantum 4.90-08
privateCA 4.90-08
emWeb 4.90-08
isclient 4.90-08
bcc 4.90-08
Snmp-Daemon-TrapLib 4.90-08
solid 4.90-08
```

### baseparamsconfig

Syntax:

**`baseparamsconfig`**

Configure Linux Base system parameters (network parameters, DNS settings, NTP settings, date and time).

### datetimeconfig

Syntax:

**datetimeconfig [-mon <1-12>] [-year <year>] [-day <1-31>[-hour <0-23>] [-min <0-59>]**

Set current system date and time.

## dnsconfig

Syntax:

**dnsconfig [options]**

Configure DNS servers.

Options are:
[-dns1 <ip>] = Primary DNS Server IP Address
[-dns2 <ip>] = Secondary DNS Server IP Address
[-dns3 <ip>] = Tertiary DNS Server IP Address

## ecnconfig

Syntax:

**ecnconfig [on,off,show,help]**

Enable, disable, or show the status of the ECN feature. You are prompted to provide the root password to execute this command.

This command takes effect until the next application restart (appstart restart).

## networkconfig

Syntax:

**networkconfig [options]**

Configure network parameters.

Options are:
[-eip <ip> ] - ELAN IP
[-egw <gw> ] - ELAN GW
[-emsk <msk>] - ELAN netmask
[-tip <ip> ] - TLAN IP
[-tgw <gw> ] - TLAN GW
[-tmsk <msk>] - TLAN netmask
[-dgw <gw> ] - Default GW
[-hn <hostname> ] - Hostname
[-fqdn <fqdn> ] - Fully qualified domain name

### ntpconfig

Syntax:

**`ntpconfig [options]`**

Configure NTP settings.

Options are:
[-cl_type <0-1>] = Clock type(0-EXTERNAL, 1-INTERNAL)
[-src_type <0-2>] = Clock source type (0-Primary,1-Secondary,2-Not a clock server) [-smode <Y/N>] = Secure mode (Y/N)
[-keyid <id> ] = NTP(in secure MD5 transfer mode) key ID
[-clIPs <ips>] = clock IPs separated by '|' symbol
[-prikey <key>] = NTP private key

### upgrade

Syntax:

**`upgrade`**

Perform backup, insert the CD, and reboot the machine.

```
$ upgrade
You have to have the root privilege to execute this command.
Password for root:
This tool will perform Linux Base upgrade.  Before the
upgrade it will back up all data.
Do you agree?  (Y/N) [n]?  y
Please insert Linux Base CD for upgrade, then press any key
```

### baseVersionShow

Syntax:

**`baseVersionShow`**

Print the base software version of the server.

```
[nortel@hp2 ~]$ baseVersionShow
nortel-cs1000-linuxbase 4.90.08
```

### ftpdisable

Syntax:

**`ftpdisable [-force/-default]`**

Disable FTP.
force = immediately stop all FTP activity
default = restore the default vsftpd configuration

### ftpenable

Syntax:

```
ftpenable [time]
```

Enable FTP. The parameter time must be a value from 1 to 20.

### ftpstatus

Syntax:

```
ftpstatus
```

Show the current FTP status.

### pins

Syntax:

```
pins [-all]
pins <patch ID>
```

Place a patch in service.

```
[nortel@arkh-linux-ibm ~]$ pins 0
Patch handle:  0 Performing the installation:
Activating patch.  Please wait...
The patch has been activated successfully
You must restart target application
```

### plis

Syntax:

```
plis
```

Show detailed information about a patch.

```
[nortel@arkh-linux-ibm ~]$ plis 0
Handle:  0
Filename:  /var/opt/nortel/patch/p11115_1.el4
Dependency List:  None
Dependency List Issue:
Patch name:  11115
Ref.  num.:  ISS1:1OF1
PRS number:  Q00000000
```

```
Engineer:  Alex Arkhipov
Release:  LINUX-4.91.12
Created:  Tue Oct 31 12:16:44 2006
Loaded:  Tue Oct 31 12:17:08 2006
Patch is out of service
Out of service date:  N/A
Patch info of element #1:
Patch name:  p11115_1_1.jar
Patch type:  JAR
Application to patch:  base:4.91.12
md5sum of the patch:  3a06b3a949012eafbe7f06f070f28d90
```

## pload

Syntax:

**`pload [-all]`**
**`pload <patch file>`**

Load the patch into the system database from the directory /var/opt/nortel/patch_directory.

pload -all loads all patches in the patch directory.
pload <patch file> loads only the specified patch.

```
[nortel@arkh-linux-ibm ~]$ pload p11115_1.el4
Patch p11115_1.el4 Patch successfully installed.
Handle:  0
```

## poos

Syntax:

**`poos [-app]`**
**`poos <patch ID>`**

Remove a patch or application from service.

```
[nortel@arkh-linux-ibm ~]$ poos 0
Patch handle:  0 Performing the uninstallation:
Deactivating patch.  Please wait...
The patch has been deactivated successfully.
You must restart target application.
```

## pout

Syntax:

**`pout <handle>`**

Unload a patch from the system database. The patch must be out of service before you can use this command.

```
[nortel@arkh-linux-ibm ~]$ pout 0
Patch 0 has been removed successfully.
```

## pstat

Syntax:

**`pstat -l, -a, <handle>`**

List installed patches.

pstat <handle> lists information for the patch with the specified handle.
pstat -l lists all installed in-service patches.
pstat -a lists all installed patches in detail.

```
[nortel@arkh-linux-ibm ~]$ pstat
In system patches:1
Patch handle 0
Filename /var/opt/nortel/patch/p11115_1.el4
Patch release version:  4.91.12
Reference number:  ISS1:1OF1
Patch is out-of-service
Patch category:  EMG
Patch special instructions:  no
Patch type:  JAR
Patch members:
p11115_1_1.jar
[nortel@arkh-linux-ibm ~]$ pstat -l
In service patches:  1
PAT# PRS/CR PATCH REF# NAME DATE FILENAME SPECINS TYPE
APPLICATION
0 Q00000000 ISS1:1OF1 p11115_1 31/10/06 p11115_1.el4 no JAR
base:4.91.12
```

## reboot

Syntax:

**`reboot`**

Reboot the system. You require a root password to use this command.

## routeconfig

Syntax:

**routeconfig [show]**
**routeconfig [add,delete] [net{network IP address},netmask{ network mask},dev{eth0/eth1/..}]**

Configure the IP routing table.

eth0 = ELAN.
eth1 = TLAN

The following example shows the output for the various uses of the routeconfig command.

```
[nortel@asa-hp4-e ~]$ routeconfig show
You have to have the root privilege to execute this command.
Password for root:
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.35.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
192.168.35.0 0.0.0.0 255.255.255.0 U 0 0 0 eth1
169.254.0.0 0.0.0.0 255.255.0.0 U 0 0 0 eth1 0.0.0.0
192.168.35.1 0.0.0.0 UG 0 0 0 eth0

[nortel@asa-hp4-e ~]$ routeconfig add -net 192.168.35.0
-netmask=255.255.255.255 -dev eth1
You have to have the root privilege to execute this command.
Password for root:
Done!

[nortel@asa-hp4-e ~]$ routeconfig show
You have to have the root privilege to execute this command.
Password for root:
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.35.0 0.0.0.0 255.255.255.255 UH 0 0 0 eth1
192.168.35.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
192.168.35.0 0.0.0.0 255.255.255.0 U 0 0 0 eth1
169.254.0.0 0.0.0.0 255.255.0.0 U 0 0 0 eth1
0.0.0.0 192.168.35.1 0.0.0.0 UG 0 0 0 eth0

[nortel@asa-hp4-e ~]$ routeconfig del -net=192.168.35.0
-netmask=255.255.255.255 -dev=eth1
You have to have the root privilege to execute this command.
Password for root:
Done!

[nortel@asa-hp4-e ~]$ routeconfig show
You have to have the root privilege to execute this command.
```

```
Password for root:
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
192.168.35.0 0.0.0.0 255.255.255.0 U 0 0 0 eth0
192.168.35.0 0.0.0.0 255.255.255.0 U 0 0 0 eth1
169.254.0.0 0.0.0.0 255.255.0.0 U 0 0 0 eth1
0.0.0.0 192.168.35.1 0.0.0.0 UG 0 0 0 eth0

Note:  When eth0 options provided ELAN gateway will be used,
eth1 - means TLAN gateway will be used.
Note:  User will be prompted to provide root password to
execute this command.
```

## swVersionShow

Syntax:

**`swVersionShow`**

Display software version of the server.

```
nortel@hp2 ~]$ swVersionShow
APP_INSTALLED=CS1000 Element Manager
APP_VERSION=4.90-08
sunAm 4.90-08
Jboss-Quantum 4.90-08
privateCA 4.90-08
emWeb 4.90-08
isclient 4.90-08
bcc 4.90-08
nortel-cs1000-linuxbase 4.90.08
Snmp-Daemon-TrapLib 4.90-08
solid 4.90-08
```

## sysbackup

Syntax:

**`sysbackup [-b,-c,-r,-s,-h]`**

Back up the system base and applications.

-b = One time backup.
-c = Add backup task to the scheduler.
-r = Remove backup task from the scheduler.
-s = Show current backup time settings.
-h = Display a help message.

### sysrestore

Syntax:

```
sysrestore
```

Restore the application to the state saved during the system backup on the specified date. When you enter this command, all applications are stopped before the restoration occurs and restart after the restoration is complete.

### faillog

Syntax:

```
faillog [-a,-r]
```

Display incorrect logon attempts (faillog -a), or reset user counters (faillog -r).

## Log messages

Linux Base logs are placed in the file /var/log/nortel/linuxbase.log. Previous log files are placed in the directory /var/log/nortel/old_logs/.

Several Linux Base subsystems exist:

- Base
- Base_Appinstall
- Base_Appstartup
- Base_Backup
- Base_Patch

To manage logging, use the baselogLevelShow and baselogLevelSet commands.

## baselogLevelShow

Syntax:

```
baselogLevelShow
```

```
[nortel@arkh-linux-ibm ~]$ /opt/nortel/base/bin/baselogL
evelShow
Subsystem Level
=============== =======
Base INFO
Base_Appinstall INFO
Base_Appstartup INFO
```

```
Base_Backup INFO
Base_Patch INFO
/opt/nortel/base/bin/baselogLevelSet
```

## baselogLevelSet

Syntax:

**`baselogLevelSet <subsystem><priority>`**

Subsystems are: Base,Base_Appinstall,Base_Appstartup,Base_Backup,Base_Patch
Priorities are: ALERT, CRIT, DEBUG, EMERG,ERROR,INFO,NONE,NOTICE,WARNING.

Nortel Communication Server 1000

# Communication Server 1000 Release 5.x Troubleshooting Guide for Distributors

Printed in Canada
Release: 5.x
Publication: NN43001-730
Document status: Standard
Document revision: 01.01
Document release date: 11 June 2008

To provide feedback or to report a problem in this document, go to www.nortel.com/documentfeedback.

www.nortel.com