Intelligent Platform Management Interface (IPMI) Conformance Test Suite (ICTS) User's Guide

intel®

Order Number A00330-002

Revision History

Revision	Revision History	Date
0.1	Prototype ICTS User’s Guide.	08/99
0.2	Prototype 2 ICTS User’s Guide.	12/99
0.3	Incorporate ICMB and IPMB Tests and Hardware Installation.	04/00
0.4	Add IPMB and ICMB transports and update batch mode setup	07/01
0.5	Added transports for IPMI 1.5, local Windows® 2000/Whistler for IA64, and SIOPROXY.EFI.	08/01
0.6	Updated text, figures, removed test descriptions and ICMB section. Add install ICTS.	01/02
0.7	Added Appendex for a description of Null Modem connectors for this iplementation	04/02
0.8	Added Command Compliance Report Using Section	12/02
0.9	Added Functional Comformance Report Section	06/03
0.10	Added Interface conformance test section	07/03
0.11	Updated for IPMI 2.0	11/04

Contents

FIGURES

TABLES

1 Introduction

The Intelligent Platform Management Interface (IPMI) initiative consists of three test areas. This User's Guide describes each test area separately:

- Section 1 provides an overview of the parts of the Conformance Test Suite (ICTS)
- Section 2 describes the installation and operation of the Platform Management Bus (IPMB)

Each section describes the steps needed to prepare for testing, to run tests, and to manage tests in progress. The available tests and their individual usage characteristics are listed.

1.1 Purpose of the IPMI Conformance Test Suite

The IPMI 1.0/1.5/2.0 Conformance Test Suite (ICTS) consists of a software framework within which you may execute custom tests or a collection of predefined pass-fail tests for determining platform conformance with the IPMI 1.0/1.5/2.0 specification. The software framework includes a graphical user interface that provides configuration, loading, execution, and response access to the tests. The interface also allows recording of configuration file information for the host environment, the platform, the target, and user preferences. Using a text editor to customize configuration files allows creation of multiple cross-platform test sets. Saved configuration files and test sets, manipulated through the graphic interface, allow multiple board and/or project conformance testing from a single host platform.

NOTE

The test suite does not imply or enforce compliance requirements.

1.2 Audience

This document supports firmware development engineers desiring IPMI 1.0/1.5/2.0 conformance. The document assumes familiarity with the IPMI 1.0/1.5/2.0 specification, engineering practices related to cross-platform development, and general testing theory. For additional information concerning the IPMI specification and cross-platform issues, refer to the *Reference Documents* section of this chapter.

1.3 ICTS Architecture Overview

The ICTS framework resides on a host system and provides a graphic interface for loading, running, and analyzing IPMI conformance tests. Once installed and configured, the system provides a set of automatically loaded tests, access to a test library, and access to library support modules for communications with transport modules.

The test framework provides an interface for loading and running tests. Tests indicate their results in two different ways, as text messages in the framework message window for human consumption, and through internal status values which are passed up from child to parent tests and eventually to the framework itself. Parent tests utilize these status values to generate summary reports of child test results in the message window.

The FTF also loads and allows tests access to message procedures from the message library. The message library facilitates sending of messages to the firmware on the target system by providing the FTF and tests with a procedure interface to transport modules to allow accurate communication access to the target.

1.3.1 ICTS Architecture Layers

The FTF is a multi-layered tool primarily implemented in the Tcl scripting language. *Transport Modules* may be either C or Tcl. The following list contains the names of the major components and descriptions of their function.

Base Framework – An application program. In combination with libraries and loaded *Test Modules* it is a complete test application. The base framework provides the user interface and the environment in which tests run.

Test Module – Conducts tests and reports a pass/fail result by conducting the test itself or by executing other test modules as sub-components. The *Base Framework* loads *Test Modules* in order to create a complete test application program.

Tool Module – Conducts procedures that do not generate pass/fail results. The Tool Module has access to the same target and framework interface resources provided to a test module.

Message Library – A set of transport-independent message passing routines using services provided by loaded *Transport Modules.* The Message Library selects the default transport in the absence of an explicit requested by the caller.

Transport Modules – A set of loadable modules, one module per transport, that provides primitives for target interface communication. The module contains additional administrative primitives not used directly by the *Test Modules*, but which are used by the *Base Framework* on behalf of the *Test Modules*.

In addition to these components, the framework includes several *Companion Libraries* that provide various utility services to the *Test Modules*.

Figure 1-1 in this section shows the relationship between the various components of the ICTS firmware test framework.

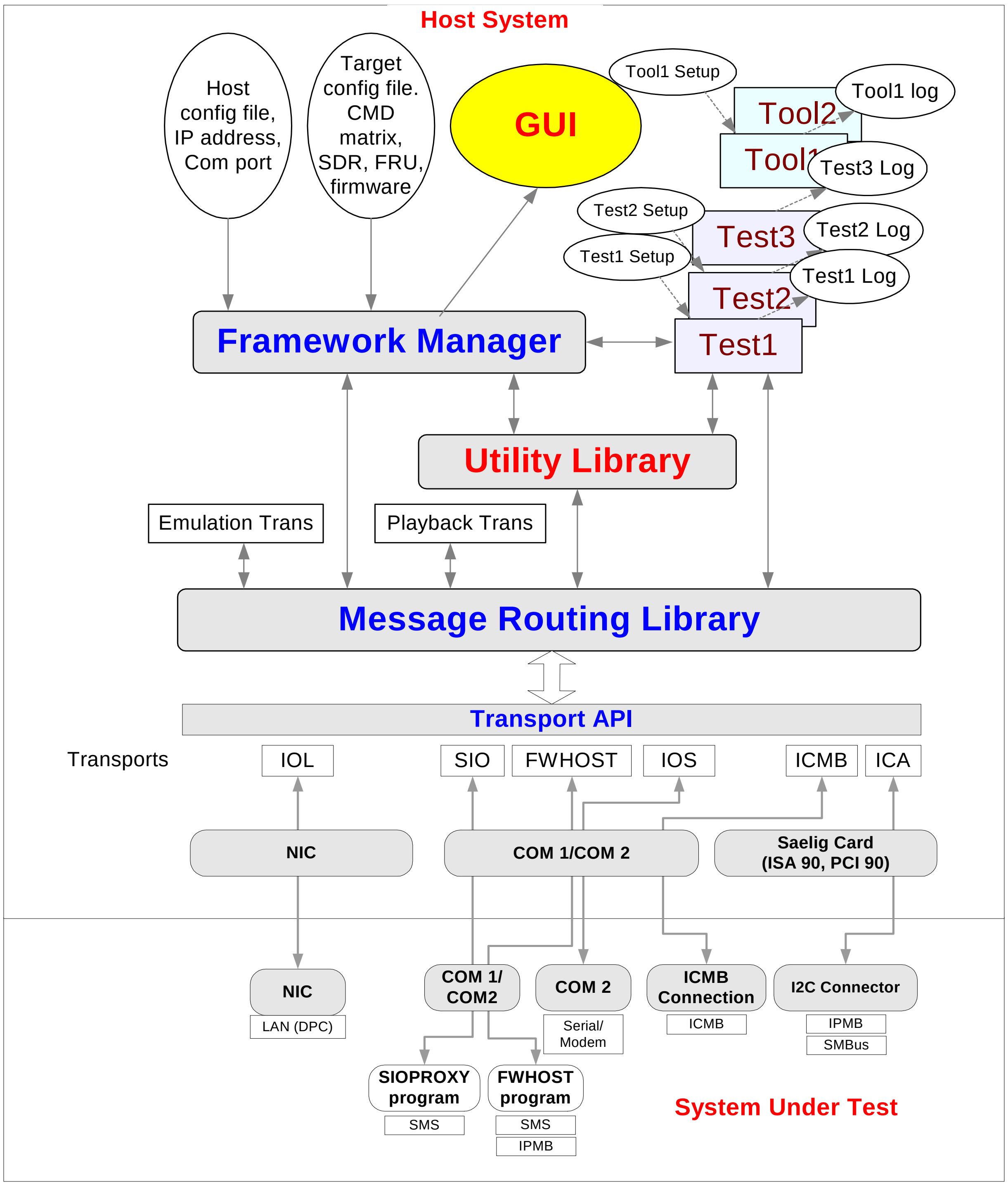


Figure 1-1. Firmware Test Framework Architecture

2 Intelligent Conformance Test Suite (ICTS)

Use of the IPMI 1.0/1.5/2.0 Conformance Test Suite assumes a host environment and a system to be tested. Before use, you must install and configure the IPMI 1.0/1.5/2.0 Conformance Test Suite (ICTS). Configurations can be saved, so configuration may not be required to run tests. Once configuration files for a host and target are complete, testing involves invoking the testing framework, loading configuration files, loading tests, running tests, and analyzing the results.

3 ICTS Installation and Configuration

Before using ICTS, you must install and configure the system. For detailed instructions on installation and configuration, refer to the *Installation and Configuration* chapter of this manual. Generally, installation and configuration involve the following steps:

1. Developing a system to be tested.
2. Installing the Intelligent Platform Management Interface Test Suite software.
3. Checking the installed directory layout.
4. Configuring the host environment.
5. Customizing the Batch Files.
6. Configuring the platform.
7. Configuring the target.
8. Optionally creating user configuration files.

3.1 Test Session Process Overview

After installation and configuration, normal testing involves the following steps:

1. Starting the framework.
2. Loading configuration files.
3. Loading tests.
4. Running the tests.
5. Analyzing the results.

3.2 ICTS Conformance Scope

ICTS provides pass-fail verification of IPMI 1.0/1.5/2.0 specifications for function module access commands and data storage requirements for SDR, SEL and FRU. The test suite does not test the internal workings of the target system.

3.3 ICTS Supports

ICTS supports the following IPMI conformance requirements:

- Management controllers, including BMC, should implement IPM functions.
- BMC should implement any mandatory functions such as IPM device, system interface, SDR repository Watchdog timer, Event Receiver, SEL Interface, and Internal Event Generator.
- Each mandatory function should be present and implemented as defined by the IPMI specification.
- Any mandatory IPMI 1.0/1.5/2.0 command should be present and implemented as defined by the IPMI specification if the specified condition from IPMI 1.0/1.5/2.0 is met or present.

- Any optional IPMI 1.0/1.5/2.0 commands should be implemented as defined by the IPMI specification if the command is present.
- Any data stored in SEL or SDR should follow the format specified in the IPMI 1.0/1.5/2.0 specifications.
- Mandatory logical devices should be implemented, such as BMC, SEL and SDR.
- If the IPMB is present, the mandatory IPMI commands should be transferred via it (unless a command is specified as mandatory for only the system interface).
- If the IPMB is present, the additional mandatory IPMB messaging support commands in the BMC must be implemented as defined by the IMPI specification.
- If the IPMB is present, it is highly recommended that the Initialization Agent should be implemented. If it is implemented then it must be done as defined by the IPMI specification.
- If LAN or Serial Channels are present, the mandatory IPMI Commands should be transferred via the respective interface using one of these channels for each LAN and Serial Channel (unless a command is specified as mandatory for only the system interface).
- The platform must provide one of the system interfaces, either KCS, SMIC or BT.

3.4 ICTS Does Not Support

ICTS does not support the following test classes:

- Target platform-specific functions are neither tested nor verified.
- The following list of functions are example of functions that do not fall under the ICTS scope:
  - OEM-specific functions. Network function code 30h-3Fh
  - Actual data field in OEM SEL Record Type C0h-DFh, Type E0h-FFh
  - Actual data field in OEM SDR, SDR Type 0C0h
  - Firmware commands. Network Function Code 08, 09
- ICTS does not measure target platform quality.
- ICTS does not do stealth tests and exception tests, unless explicitly mentioned in the IPMI 1.0/1.5/2.0 specification.
- ICTS does not detect management controllers in the target system. It only takes target system information from the SDR.

For additional information on the IPMI 1.0/1.5/2.0 specification, refer to the specification listed in the *Reference Documents* section of this chapter.

3.5 Reference Documents

In addition to the information in this manual, the following documents may provide information useful to testing for IPMI 1.0/1.5/2.0 conformances:

- *Intelligent Platform Management Interface Specification* v2.0 Revision 1.0, © 2004 Intel® Corporation, Hewlett-Packard Company, NEC Corporation, and Dell Computer Corporation.
- *Intelligent Platform Management Interface Specification* v1.5 Revision 1.0, © 2001 Intel® Corporation, Hewlett-Packard Company, NEC Corporation, and Dell Computer Corporation.
- *Intelligent Platform Management Interface Specification* v1.0 Revision 1.1, © 1999 Intel® Corporation, Hewlett-Packard Company, NEC Corporation, and Dell Computer Corporation.

- *Intelligent Platform Management Bus Communications Protocol Specification* v1.0, rev. 1.2 © 2000 Intel® Corporation.
- *Intelligent Chassis Management Bus Bridge Specification v1.0,* rev. 1.2, © 2000 Intel® Corporation.
- *System Management Bus (SMBus) Specification, Version 2.0,* ©2000, Duracell Inc., Fujitsu Personal Systems Inc., Intel® Corporation, Linear Technology Corporation, Maxim Integrated Products, Mitsubishi Electric Corporation, Moltech Power Systems, PowerSmart Inc., Toshiba Battery Co., Ltd., Unitrode Corporation, USAR Systems.

3.6 Intel® Glossary

This section contains a terms used throughout this document. For additional information regarding terms, refer to the documents listed in the *Reference Documents* section of this manual.

API	Application Program Interface.
EFI	Extensible Firmware Interface
FTF	Firmware Test Framework.
FRU	Field Replaceable Unit. A module or component that will typically be replaced in its entirety as part of a field service or repair operation.
Host	The machine executing the test, which may or may not be the same as the target.
ICTS	Intelligent Platform Management Interface (IPMI) Conformance Test Suite.
Interface	The local communication path on the target machine (I2C, SMS, etc.)
IPMB	Intelligent Platform Management Bus. Name for the architecture, protocol, and implementation of a special bus that interconnects the baseboard and chassis electronics and provides a communications medium for system platform management information. The bus is built on I^2C and provides a communications path between management controllers such as the BMC, the ICMB bridge controller, and the chassis management controller.
IPMI	Intelligent Platform Management Interface.
SDR	Sensor Data Record. A data record that provides platform management sensor type, locations, event generation, and access information.
Target	The machine under test, which may or may not be the same as the host.
Transport Layer	The data communication path between the host and target machines (Local, RS-232, TCP/IP LAN, etc.).
UI	User Interface
IPM	Intelligent Platform Management
BMC	Baseboard Management Controller
SEL	System Event Log
Saelig card	A card providing the standard I2C interface to access the IPMB

3.7 ICTS User Interface

The Intelligent Platform Management (IPMI) Conformance Test Suite (ICTS) consists of a host-executable graphical user interface to allow configuration of conformance tests, running of tests, and viewing of test results. The interface consists of a message window, a status bar, and menus. This chapter provides an overview description of the interface window and menus. For information on launching the test framework and on using the dialog boxes for configuration, testing and analysis, refer to the Installation and Configuration chapter of this manual and to the

Tutorial chapter of the *Intelligent Platform Management (IPMI) Conformance Test Suite (ICTS) Developer's Guide*.

3.8 Windows

Each window or menu has a specific purpose. A description of each appears in this chapter, and every section contains a description of the window or menu and provides a graphic of its appearance during typical use.

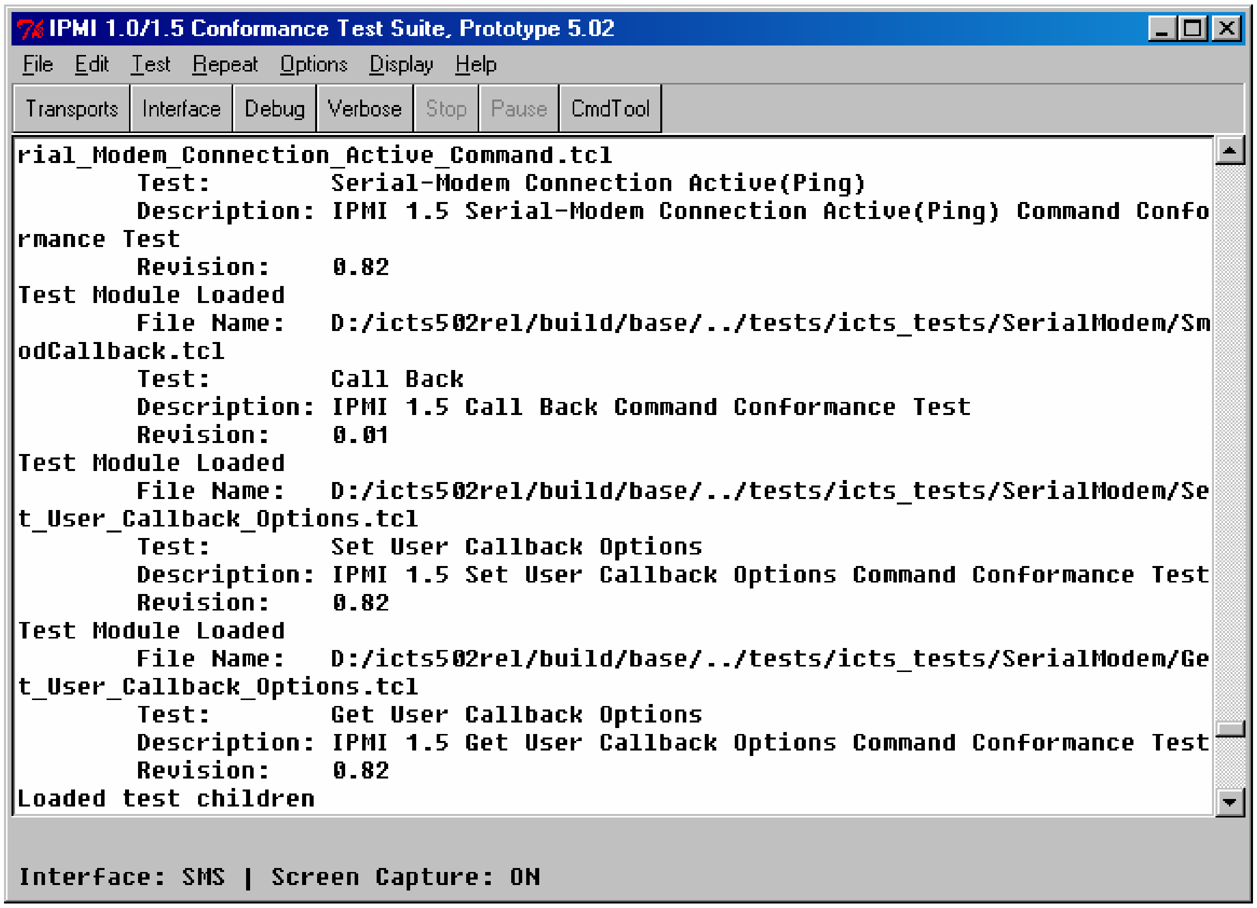
Table 3-1 contains a list of the windows and their purposes.

Table 3-1. ICTS Graphical User Interface Windows

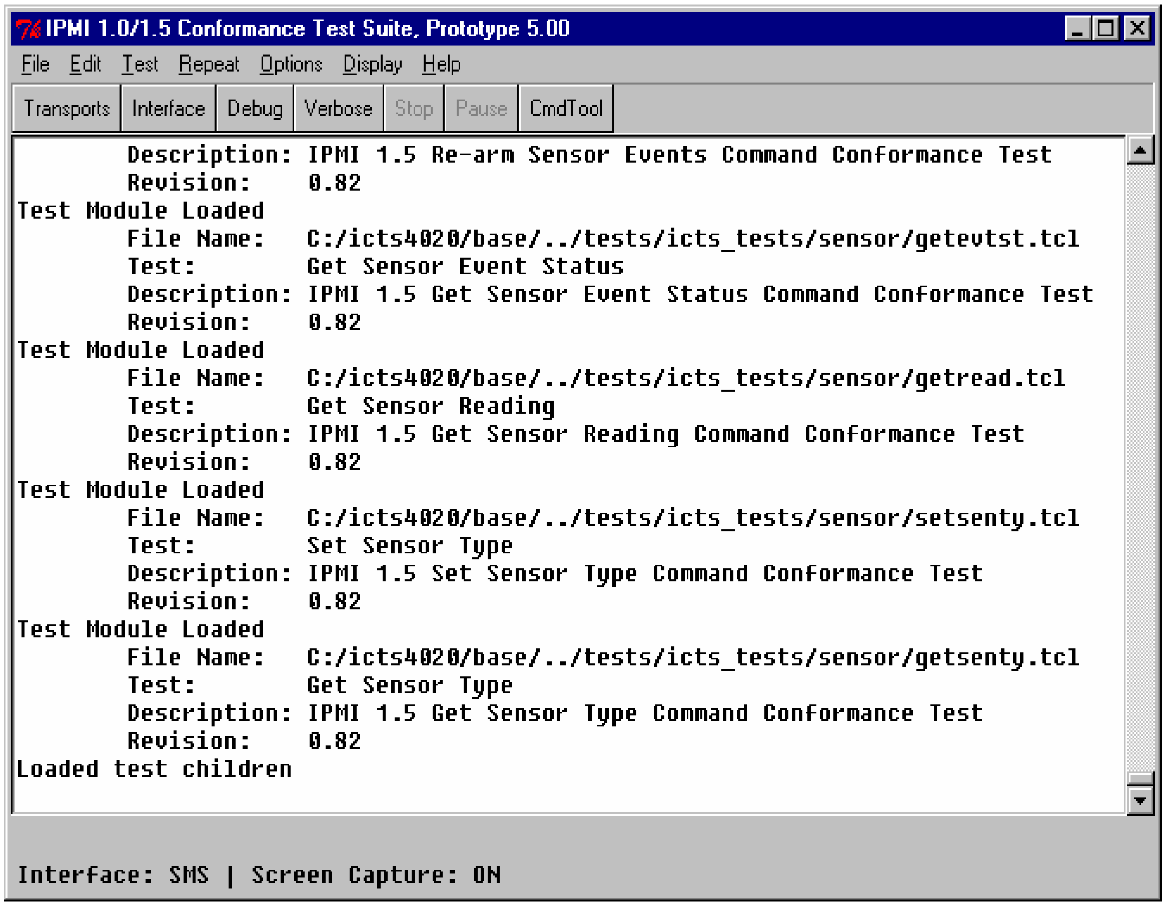
Screen Object Name	Purpose
Main Message Window	Provides space for viewing messages generated by configuration, loading, testing, and analysis. The menu bar at the top provides access to the other menus.
Menu Bar	Provides access to drop-down menus.
Status Bar	Displays messages from test or from the framework.

3.8.1 Main Window

The main ICTS window provides access to menus, system information, and test feedback. Figure 5-1 depicts the main window immediately after launch. Diagnostic messages for configuration file-loaded tests have scrolled past, and the status bar at the bottom of the window shows the framework’s initial state.



```
rial_Modem_Connection_Active_Command.tcl
        Test:         Serial-Modem Connection Active(Ping)
        Description: IPMI 1.5 Serial-Modem Connection Active(Ping) Command Confo
rmance Test
        Revision:     0.82
Test Module Loaded
        File Name:    D:/icts502rel/build/base/../tests/icts_tests/SerialModem/Sm
odCallback.tcl
        Test:         Call Back
        Description: IPMI 1.5 Call Back Command Conformance Test
        Revision:     0.01
Test Module Loaded
        File Name:    D:/icts502rel/build/base/../tests/icts_tests/SerialModem/Se
t_User_Callback_Options.tcl
        Test:         Set User Callback Options
        Description: IPMI 1.5 Set User Callback Options Command Conformance Test
        Revision:     0.82
Test Module Loaded
        File Name:    D:/icts502rel/build/base/../tests/icts_tests/SerialModem/Ge
t_User_Callback_Options.tcl
        Test:         Get User Callback Options
        Description: IPMI 1.5 Get User Callback Options Command Conformance Test
        Revision:     0.82
Loaded test children

Interface: SMS | Screen Capture: ON
```


IPMI 1.0/1.5 Conformance Test Suite, Prototype 5.00
File Edit Test Repeat Options Display Help
Transports Interface Debug Verbose Stop Pause CmdTool

```
        Description: IPMI 1.5 Re-arm Sensor Events Command Conformance Test
        Revision:    0.82
Test Module Loaded
        File Name:   C:/icts4020/base/../tests/icts_tests/sensor/getevtst.tcl
        Test:        Get Sensor Event Status
        Description: IPMI 1.5 Get Sensor Event Status Command Conformance Test
        Revision:    0.82
Test Module Loaded
        File Name:   C:/icts4020/base/../tests/icts_tests/sensor/getread.tcl
        Test:        Get Sensor Reading
        Description: IPMI 1.5 Get Sensor Reading Command Conformance Test
        Revision:    0.82
Test Module Loaded
        File Name:   C:/icts4020/base/../tests/icts_tests/sensor/setsenty.tcl
        Test:        Set Sensor Type
        Description: IPMI 1.5 Set Sensor Type Command Conformance Test
        Revision:    0.82
Test Module Loaded
        File Name:   C:/icts4020/base/../tests/icts_tests/sensor/getsenty.tcl
        Test:        Get Sensor Type
        Description: IPMI 1.5 Get Sensor Type Command Conformance Test
        Revision:    0.82
Loaded test children

Interface: SMS | Screen Capture: ON
```

Figure 3-1. IPMI Conformance Test Suite Main Window

3.8.2 Status Bar

The Status Bar at the bottom of the main window shows general information about the user interface, including the target configuration, and the status of the screen capture utility. Additional information appears according to changes in the status of the framework.

3.8.3 Menus

Drop-down menus conform to Microsoft† Windows NT† graphical user interface specifications. Each menu has a title in the menu bar at the top of the main window area. Menu commands are organized by type under menu titles. Click and hold on the menu title to access the menu items. Positioning the cursor over the menu command in the drop-down list and releasing the mouse button invokes the command item. Menu items followed by ellipses (. . .) invoke dialog boxes.

Every menu and menu command has a corresponding keystroke. Invoke drop-down menus by pressing the *ALT* key plus the underlined letter of the menu title. For example, to display the **File** drop-down menu, hold *ALT* and press *F*. Once a menu is invoked, pressing the underlined letter of any command listed in the drop-down executes that command. The *ESC* key cancels drop-down access.

This chapter contains a section describing each drop-down menu and its commands. Each section shows the screen during normal use. Table Table 3-2 shows the menu titles, their purposes, and the keystrokes for access.

Table 3-2. ICTS Graphical User Interface Menus

Menu Name	Keystroke	Purpose
File	ALT+F	Loading and saving configuration files.
Edit	ALT+E	Copying selected text.
Test	ALT+T	Loading, pausing, and stopping a test module.
Repeat	ALT+R	Repeating the last test run or the last tool used.
Options	ALT+O	Configuring debug levels, message routing, interface type, data source, and microcontroller type.
Display	ALT+D	Configuring window content and appearance.
Help	ALT+H or F1	Providing support information.

3.8.3.1 File Menu

The File drop-down menu gives access to dialog boxes that let you load configuration files for the host environment, the platform, the target, and the user. Additionally, the **File** menu offers access to a dialog box for modifying and saving user configuration files. Figure 3-2 shows the contents of the **File** drop-down menu:

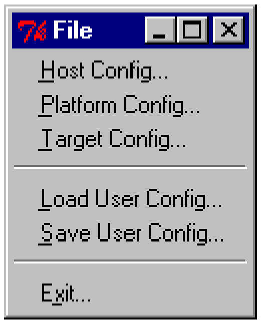


Figure 3-2. File Menu

The following list contains the menu items offered in the **File** menu along with a function description of each:

- Host Config...—Presents a dialog box for loading a host configuration file. Figure 3-3 shows the Host Config... dialog box:

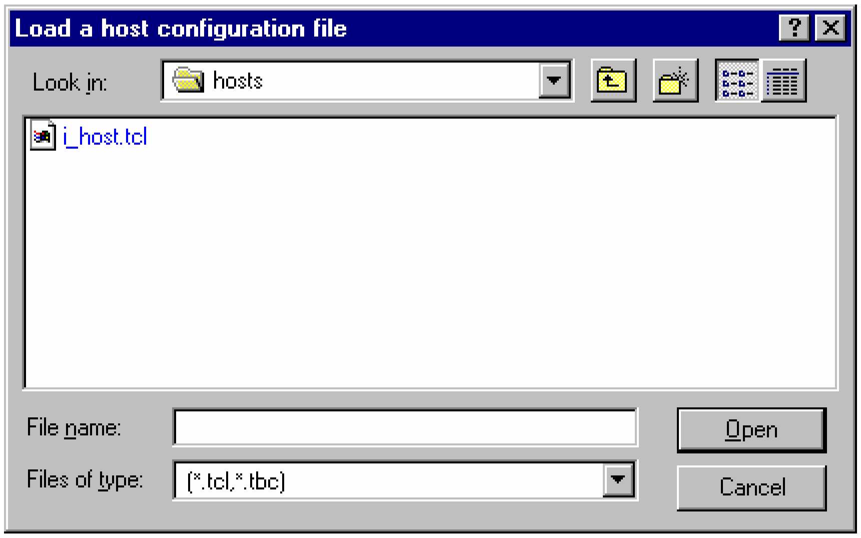


Figure 3-3. Load a Host Configuration File Dialog

- Platform Config...—Presents a dialog box for loading a platform configuration file. This menu item remains unavailable until a host configuration file is successfully loaded. Except in name, the Platform Config... dialog box is identical to the box shown in Figure 3-3.
- Target Config...—Presents a dialog box for loading a target configuration file. This menu item remains unavailable until a platform configuration file is successfully loaded. Except in name, the Target Config... dialog box is identical to the box shown in Figure 3-3.
- Load User Config...—Presents a dialog box for loading a user configuration file. This menu item remains unavailable until a host configuration file is successfully loaded. Except in name, the Load Config... dialog box is identical to the box shown in **Error! Reference source not found.**.

- Save User Config...—saves the current users configurations into the file i_user.tcl. This menu item remains unavailable until a host configuration file is successfully loaded. **Error! Reference source not found.** shows the Save User Config... dialog box:
- Exit—Exits from the program after presenting a confirmation dialog. Figure 3-4 shows the Exit... dialog box:

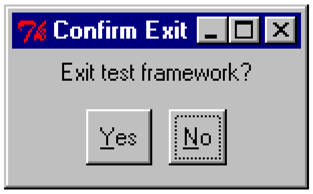


Figure 3-4. Exit Framework Confirmation Box

3.8.3.2 Edit Menu

The Edit drop-down menu contains only the Copy command item. The Copy item remains unavailable, as shown in Figure 3-6, until a block of text is selected. Once text is selected, the Copy item appears as bold and enabled, as shown in Figure 3-5. Selecting the Copy item places the currently selected text in the system clipboard.

Figure 3-5. Edit Menu Copy-enabled

Figure 3-6. Edit Menu Copy-disabled

3.8.3.3 Test Menu

The Test drop-down menu has a unique characteristic. After a test is loaded, an additional menu item representing the newly loaded test appears in the IPMI 1.0/1.5/2.0 Conformance Test Suite drop-down menu. For additional information on using the Test drop-down menu to load, start, and stop tests, refer to *Preparing to Run Tests* chapter of this manual.

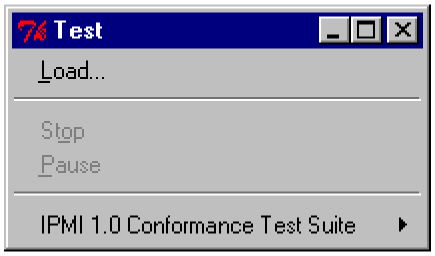


Figure 3-7. Test Menu

Initially, the Test drop-down menu contains these items:

- Load. . .—Presents a dialog to allow selection and loading of a test module. This menu item remains dimmed until a target configuration file (or in some cases only a platform configuration file) is successfully loaded. For each loaded test, an item is added to the test menu. Figure 3-8 shows the Test menu with a custom built test named, "*My Custom Test*" loaded:

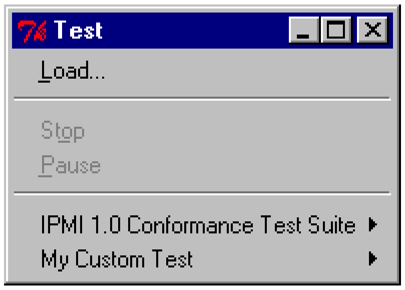
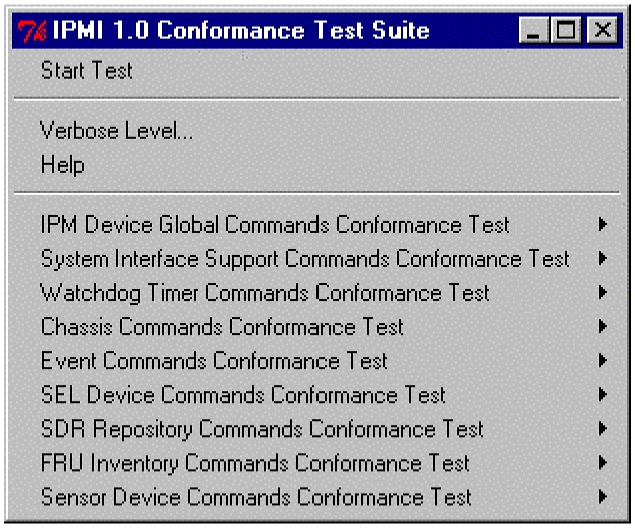


Figure 3-8. Test Menu Showing Loaded Custom Test

- Stop—Stops the currently running test or tool, if possible. This menu item is unavailable unless a test or tool is currently running.
- Pause—Pauses the currently running test or tool, if possible. This menu item is dimmed unless a test or tool is currently running.

- IPMI 1.0/1.5/2.0 Conformance Test Suite—Provides access to additional drop-down menus for configuration of tests and access to test data. Figure 3-9 shows the contents of the IPMI 1.0/1.5/2.0 Conformance Test Suite dialog box. The dialog provides access to a series of drop-down menus through which you can access the automatically loaded test libraries provided by the framework. For additional information on the drop-down menus in this dialog, refer to the *IPMI Conformance Test Suite Sub-Menus* section of this chapter.


IPMI 1.0 Conformance Test Suite
Start Test
Verbose Level...
Help
IPM Device Global Commands Conformance Test
System Interface Support Commands Conformance Test
Watchdog Timer Commands Conformance Test
Chassis Commands Conformance Test
Event Commands Conformance Test
SEL Device Commands Conformance Test
SDR Repository Commands Conformance Test
FRU Inventory Commands Conformance Test
Sensor Device Commands Conformance Test

-

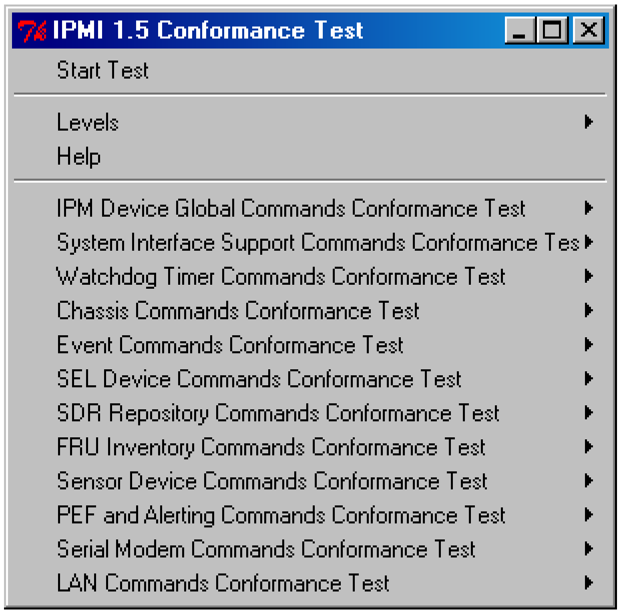

IPMI 1.5 Conformance Test
Start Test
Levels
Help
IPM Device Global Commands Conformance Test
System Interface Support Commands Conformance Tes
Watchdog Timer Commands Conformance Test
Chassis Commands Conformance Test
Event Commands Conformance Test
SEL Device Commands Conformance Test
SDR Repository Commands Conformance Test
FRU Inventory Commands Conformance Test
Sensor Device Commands Conformance Test
PEF and Alerting Commands Conformance Test
Serial Modem Commands Conformance Test
LAN Commands Conformance Test

Figure 3-9. IPMI 1.0/1.5/2.0 Conformance Test Suite Drop-Down Menus

IPMI Conformance Test Suite Sub-Menus

The IPMI 1.0/1.5/2.0 Conformance Test Suite submenus give access to the testing libraries provided by the test framework. Clicking Start Test on the IPMI 1.0/1.5/2.0 Conformance Test Suite menu runs the entire set of loaded tests. Selecting a child test from the test list sub-menu provides access to another test drop-down menu. The new menu appears identical to the one shown in Figure 3-9, except that the list of child tests changes. Each child test menu item provides access to automatically loaded tests. Selecting their respective Start Test item will run that test and its children.

Every test drop-down provides its own Start Test item, as well as Levels and Help items. Levels allows access to debug and verbose level-setting. Help provides information available for the currently loaded tests.

Following the test hierarchy down to a test with no children in the sub-menu allows execution of a single test without initiation of child tests. Figure 3-10 shows the Get Device ID test menu. Get Device ID has no child tests, so there are no additional menu items.

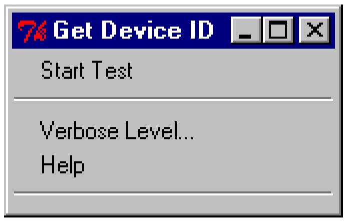


Figure 3-10. Standard Test Dialog

NOTE

The menus for a Custom Test are identical in form and content to the menus and actions for the standard tests described in this section.

3.8.3.4 Repeat Menu

The Repeat drop-down menu allows quick re-invocation of the last test run. Figure 3-11 shows the Repeat menu before a test has been run:

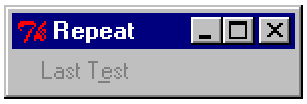


Figure 3-11. Repeat Menu

The Repeat drop-down menu contains these items:

- Last Test—Repeats the most recently completed test. This item remains dimmed until at least one test has successfully completed. During normal operation, the name of the last test run appears in this menu.

After a test has run, the Last Test item is enabled and shows the name of the last test run. Figure 3-12 shows the Repeat menu after the Get Device ID test has run:

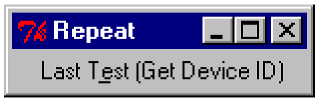


Figure 3-12. Repeat Menu Showing Name of Last Test Run

3.8.3.5 Options Menu

The ICTS Options menu allows customization of the framework user interface behavior. For detailed usage information about the options presented by this menu, refer to the *Installation and Configuration* chapter in this manual.

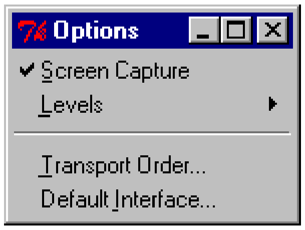


Figure 3-13. Options Menu

The Options drop-down menu contains the following items:

- Screen Capture—Toggles logging of screen messages to the host log file.
- Levels—Displays a cascading menu containing the Debug and Verbose items. Either item presents a dialog box for setting the global level for the selected item. Except for the label, the dialog box for both items appears as shown in Figure 3-15. Figure 3-14 shows the Levels menu:

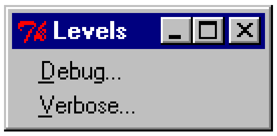


Figure 3-14. Levels Menu Items

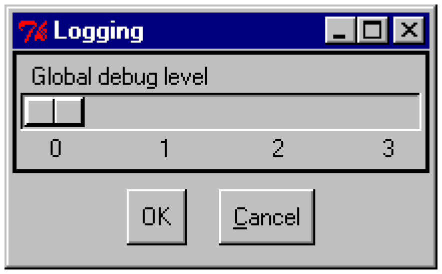


Figure 3-15. The Level Dialog (Global Debug)

-

Transport Order. . .—Presents a dialog box for changing the priority order when multiple transport modules are available. Clicking on an item in the dialog promotes that item to the highest priority. Figure 3-16 shows the Transport Module Order dialog box containing two available transport layers:

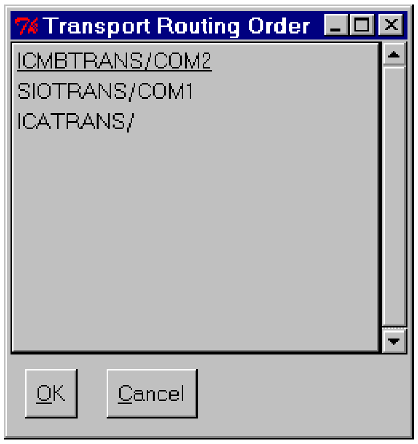


Figure 3-16. Transport Module Order Dialog Box

- Default Interface ...—Presents a dialog box for selecting the default target interface. This dialog lists all open interfaces. Clicking on an item in the list selects that interface. Clicking OK makes the selected interface the default interface. The result appears in the status bar on the main window. Figure 3-17 shows the Default Interface dialog:

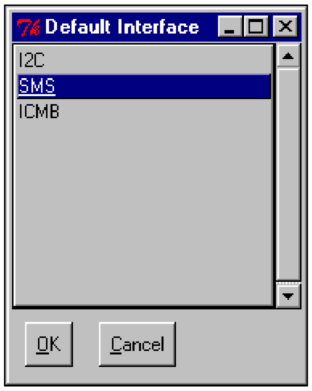


Figure 3-17. Default Interface Dialog Box

3.8.3.6 Display Menu

The Display menu items control the appearance of the ICTS interface window by allowing changes to the appearance of the graphical interface. Figure 3-18 shows the Display drop-down menu:

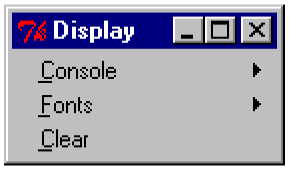


Figure 3-18. Display Menu

The Display drop-down menu contains these items:

- Console—Provides access to the Console menu. The Show option brings up the wish console window (a built-in feature of Tcl/Tk on Win32†). The Hide option makes the window disappear. The window allows you to enter Tcl commands. For additional information on use of the Console window, refer to your Tcl/Tk documentation. Shows the Console menu:

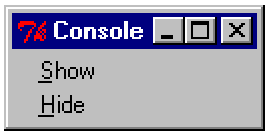
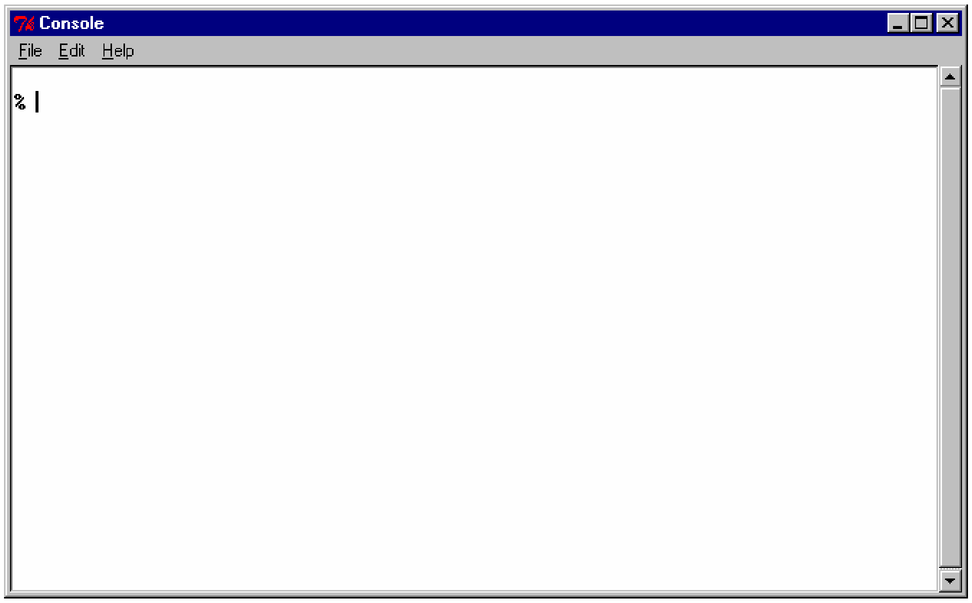


Figure 3-19. Console Menu

Figure 3-20 shows the Console window:



```
% |
```

Figure 3-20. Console Window

-

Fonts—A cascading menu that provides access to font family selection and font size selection dialog boxes. Selecting the Default item returns the interface to its original state. Figure 3-21 shows the Fonts dialog box:

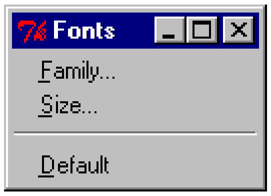


Figure 3-21. Fonts Dialog Box

Selecting the Family. . . item brings the Font Families dialog box up. Selecting a font family from the dialog and clicking the OK changes the font family for the user interface. Figure 3-22 shows the Font Families selection dialog box:

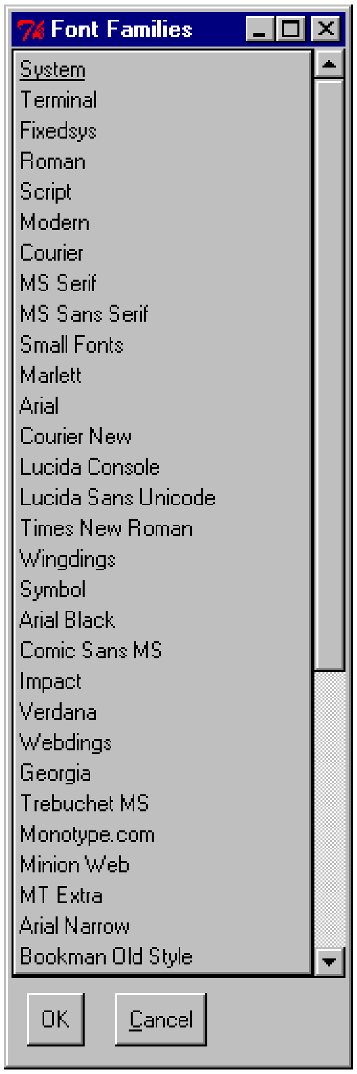


Figure 3-22. Font Families Selection Window

Selecting the Size. . . item brings the Font Sizes dialog box up. Selecting a size from the dialog and clicking the OK changes the font size for the user interface. Figure 3-23 shows the Font Sizes selection dialog box:

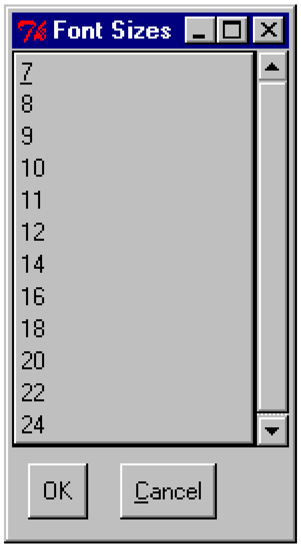


Figure 3-23. Font Sizes Selection Window

3.8.3.7 Help Menu

ICTS provides context-sensitive help through the F1 key and the Help drop-down menu. Shows the Help drop-down menu:

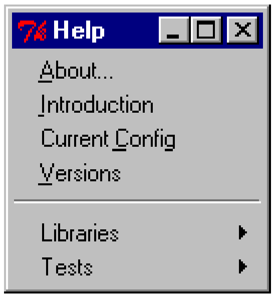


Figure 3-24. Help Menu

The Help menu contains these items:

- About. . .—Presents a window displaying the program name, version number and copyright information.
- Introduction—Displays a brief how-to-get-started tutorial in the message window. The tutorial appears automatically if the framework starts without a host configuration file.
- Current Config—Displays the current value of all host, platform, target, and user configuration variables in the message window.
- Versions– Displays version number information for various components of the framework in the message window.

- Libraries—Displays a cascading list of loaded libraries. Selecting an item from the list displays help for that item. The list contents depend on the libraries currently loaded.
- Tests—A cascading menu containing an entry for each loaded test module. Selecting an entry displays information about the corresponding test in the message window.

4 Installation and Configuration

This chapter contains information on transport modules as well as hardware and software requirements for the IPMI Conformance Test Suite (ICTS). It also contains ICTS installation and configuration procedures along with the hardware and software requirements for ICTS.

4.1 ICTS Operating Requirements

Your Intelligent Platform Management Interface (IPMI) Conformance Test Suite (ICTS) requires hardware and software components. In addition, you will need to access the Intel® IPMI Web site and download an executable .zip file containing installation and configuration files. This section contains descriptions of the components you will need to install and configure the test suite.

4.1.1 Hardware Requirement

ICTS requires access to Tcl/Tk software, transport modules, configuration files, and hardware. Before attempting to install the Test Suite, check your hardware to be sure the minimum requirements listed below are met:

- IA microprocessor
- 24MB of available system memory (32MB is recommended)
- 20MB of disk space
- VGA adapter and monitor capable of providing 16 color, 640x480 graphics mode
- 101-key or compatible keyboard
- Mouse (optional)
- RS485 / Converter (ICMB)
- Saelig Interface card (IPMB)
- Second Saelig Interface card (SMBus) (optional only if IPMB or SMBus is used)
- Network Interface Card (LAN)
- Appropriate cable connections

4.1.2 Software Requirements

Before installing the ICTS software, be sure the following requirements are met:

- Installed Windows† 95, Windows 98, Windows NT 4.0, Windows® 2000 or Windows® 2000 Whistler for Itanium™ brand servers.
- Tcl/Tk 8.3.0 through 8.3.9. (The use of different versions will cause unpredictable behavior.)

4.2 Transport Modules

Transport modules are Tcl or C routines that allow messaging across a particular transport layer. Transport modules share an API. Some modules do not implement every standard routine, and some modules contain non-standard routines. At least one transport module must be present to run tests.

WARNING

Avoid direct calls to transport module routines. Use the Message Library or higher-level libraries described in the *Intelligent Platform Management Interface (IPMI) Conformance Test Suite (ICTS) Developer's Guide*.

4.2.1 General Requirements

Transport modules are passive Tcl or C routines that require no other procedure or data definitions before loading. They do not initiate interactions with their environment. Transport routines require a Tcl interpreter, but do not require the ICTS environment to load. Once loaded, the ICTS framework automatically determines the supported commands and interfaces of a loaded transport module.

4.2.2 Local Transport Module

The LOCTRANS implementation is a driver *imbdrv.sys* that can be installed under Windows NT 4.0 with service pack #4 or greater. The driver can be installed under Windows® 2000. The driver can be installed under Windows® 2000 Whistler for Itanium™ brand servers, though the driver handles the Windows® 2000 Whistler environment the ICTS application runs under the 32-bit application environment. The following Table 4-1 lists the cross-layer communication and the LOCTRANS implementation for the Local Transport Module:

Table 4-1. LOCTRANS Implementation

Transport Characteristic	LOCTRANS Implementation
Software implementation	Four files: `loctrans.tcl`, `imb.dll`, `imbapi.dll`, and `imbdrv.sys`. The latter two files are part of the standard server-management software stack
Logical transports	None
IPMI interfaces	SMS
IPMI interface nodes	None
Raw interfaces	None
Non-IPMI interfaces	None
Target hardware requirements	KCS or SMIC interface
Target software requirements	The `imbdrv.sys` driver must be installed under Windows NT 4.0 with service pack #4 or Windows® 2000
Host hardware requirements	Must be the same physical machine as the target
Host software requirements	None
Host operating system	See OS requirement under "Software Requirements"
Tcl/Tk version	Tested with 8.3.0 through 8.3.9
`topen` portid	None
`topen` options	None
`topen` limitations	Only one open port is allowed at any given time
`tsend/tget` limitations	Limited only by available memory
Incoming queue limitations	Limited only by available memory

Timeout implementation	The timeout value is the ISA timeout between the IMB driver and the firmware. The default is 300ms and it may be changed with `ttimeout`. The timeout parameter to `tget` and `trawget` has no meaning.
Option procedures	`terror, tdebug`
Non-standard procedures	None

4.2.3 The Serial I/O (SIO) Transport Module

The following Table 4-2 lists the cross-layer communication characteristics and SIOTRANS implementation for the SIO Serial Transport Module:

Table 4-2. SIOTRANS Implementation

Transport Characteristic	SIOTRANS Implementation
Software Implementation	Single file, siotrans.tcl
Logical Transports	None
Interface Types	IPMI, CFG
Interfaces (IPMI)	SMS
Interfaces (CFG)	SIO (see below)
Interface Nodes	None.
Raw Interfaces	None
Target Hardware Requirements	Null modem cable connected to COM port.
Target Software Requirements	Must be running SIOPROXY.EXE under DOS or SIOPROXY.EFI under the EFI Shell.
Host Hardware Requirements	Other end of null modem cable connected to a COM port.
Host Software Requirements	None.
Host Operating System	Win9x, WinNT, Win2000
Tcl/Tk Version	Tested with 8.3.0 through 8.3.9
Topen portid	COM port name (COM1, COM2, etc.). No default.
Topen options	None
Topen limitations	None
Tsend/tget limitations	None.
Incoming queue limitations	None.
Timeout Implementation	Default value is 1 second, but set in I_tranget.tcl to 6 seconds. May be changed with ConfigTool (see below).
Optional Procedures	tdebug, tversion
Non-standard Procedures	None.

NOTE

See Appendix for Null Modem description and layout.

SIO Interface

The “SIO” interface is a non-IPMI interface that can be used to send configuration commands to the SIOPROXY program using `tsend` and `tget`. A cooked SIO command for `tsend` has the form:

```
<configuration cmd> <data1> <data2> ... <dataN>
```

The commands are listed in the Table 4-3. The response format from `tget` for the SIO interface is:

<data1> <data2> ... <dataN>

Table 4-3. SIO Configuration Commands

Code	Command	Request Data	Response Data
00h	Hello	-	Byte 1: Completion Code Byte 2: Version Number (LSB) Byte 3: Version Number (MSB)
01h	Exit Program	-	Byte 1: Completion Code
02h	Verbose Level	Byte 1: New Verbose Level (optional)	Byte 1: Completion Code Byte 2: Previous Verbose Level
03h	Debug Level	Byte 1: New Debug Level (optional)	Byte 1: Completion Code Byte 2: Previous Debug Level
04h	Text Message	Bytes 1-N: A null terminated text message to be displayed on the screen of the target machine.	Byte 1: Completion Code

Debug Levels

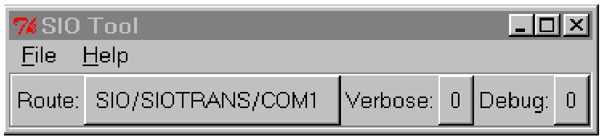
The debug levels for the SIOTRANS implementation of `tdebug` are as follows:

Level 0: No messages.

Level 1: Misc. debug messages.

Level 2: Print all outgoing and incoming raw data plus the level 1 messages.

The Verbose and Debug Levels SIO configuration commands can be changed using the SIO Tool. The tool can be loaded into ICTS from the tests\cfgtools folder and is called Sio Tool.tcl. Figure 4-1 shows the SIO tool:



Route: SIO/SIOTRANS/COM1 Verbose: 0 Debug: 0

Figure 4-1. SIO Tool

4.2.4 The IPMB Transport Module

The following Table 4-4. lists the cross-layer communication characteristics and ICATRANS implementation for the IPMB Transport Module:

Table 4-4. IPMB Implementation

Transport Characteristic	ICA90 and PCI90 (Saelig) Implementation
Software Implementation	Four files: `icatrans.tcl, icadrv.dll` and `icaio.sys/pci_i2c.sys`.
Logical Transports	None.
Interface Types	IPMI, CFG
Interfaces (IPMI)	I2C
Interface (CFG)	(CFG)
Raw Interfaces	I2C-Fully supported. The interface uses polled mode.
Target Hardware Requirements	None.
Target Software Requirements	None.
Host Hardware Requirements	ICA90(Saelig) ISA or PCI90(Saelig) PCI card with IPMB cable connected to the target system.
Host Software Requirements	None.
Host Operating System	WinNT, Win2000
Tcl/Tk Version	Tested with 8.3.0 through 8.3.9
`topen` portid	ICA90 card base address (default address is 0x310) or 0-based PCI90 card number.
`topen` options	ICA90 or PCI90 card's I2C slave address. Default slave address is 0x54.
`topen` limitations	One open port at a time.
`tsend/tget` limitations	Message queue handles up to 64(0x3F) messages.
Incoming queue limitations	Message queue handles up to 64(0x3F) messages.
Timeout Implementation	Default value is 1 seconds, but set in I_tranget.tcl to 6 seconds. May be changed with ConfigTool (see below).
Optional Procedures	`terror, tdebug`
Non-standard Procedures	None.

Debug Levels

The debug levels for the ICATRANS implementation of `tdebug` are as follows:

Level 0: No messages.

Level 1: Print all incoming cooked or raw data based on the type of call.

Level 2: Print all outgoing cooked or raw data based on the type of call, plus level 1 messages.

Level 3: Print all outgoing and incoming, i.e., level 1 messages plus level 2 messages.

4.2.5 The ICMB Transport Module

The following Table 4-5. lists the cross-layer communication and the ICMBTRANS implementation for the ICMB Transport Module:

Table 4-5. ICMBTRANS Implementation

Transport Characteristic	ICMBTRANS Implementation
Software Implementation	Single file, icmbtrans.tcl
Logical Transports	None
Interface Types	IPMI, CFG
Interfaces (IPMI)	ICMBTRANS: ICMB
Interfaces (CFG)	CFG
Interface Nodes	Corresponds to target ICMB Address specified in hexadecimal (e.g. 0x8019)
Raw Interfaces (IPMI)	ICMBTRANS: None
Target Hardware Requirements	ICMBTRANS: ICMB connector
Target Software Requirements	ICMBTRANS: Target must be configured for the specified COM port (Refer to configuring to a specific COM port for Quatech card)
Host Hardware Requirements	Other end of the spliced cable is connected to D9 connector of the RS485 converter. Tx (A) and Rx (A) must be wired together, Tx(B), Rx(B) must also be wired together
Host Software Requirements	None.
Host Operating System	Win9x, WinNT, Win2000
Tcl/Tk Version	8.3.0 through 8.3.9
`Topen` portid	COM port name (COM1, COM2, etc.) No default.
`Topen` options	Node Address specified in hexadecimal (e.g. 0x1000). May be changed with ConfigTool.
`Topen` limitations	One open port at a time.
`Tsend` limitations	If iface parameter is to ICMB interface, it must include the target node address. For example, ICMB:0x8019
`Tget` limitations	Does not allow receiving broadcast packets.
Incoming queue limitations	Only packets targeted to the host ICMB node will be moved into the queue
Timeout Implementation	Default value is 1 seconds, but set in I_tranget.tcl to 6 seconds. May be changed with ConfigTool (see below).
Optional Procedures	Tdebug
Non-standard procedures	None.

4.2.5.1 Specifications and Characteristics

The ICMB transport module allows three levels of debug through `tdebug`. Setting debug to zero disallows messages. Setting debug to 2 sends all outgoing and incoming raw data to the port.

Debug Levels

The debug levels for the ICMBTRANS implementation of `tdebug` are as follows:

Level 0:	No messages.
Level 1:	No messages. (Reserved for future used.)
Level 2:	Print all outgoing and incoming raw data plus the level 1 messages.

ICMB Raw Messages

The trawsend/trawget procedures are not implemented by the ICMB transport.

Cooked Message Format

The "ICMB" is an IPMI interface and hence the cooked message format of ICMB is identical to the IPMI cooked message format with the following interpretation of the NetFn field to support bridging:

All requests to the ICMB interface requests are bridged with one exception. Any requests that are sent using the 'Bridge' Net Function are assumed to be directed at the bridge device it self and will be routed appropriately.

<rsSA> <NetFn> <rsLUN> <Cmd> <Data1> <Data2> … <DataN>

where:

rsSA	Responder's I2C Slave Address
NetFn	Network function, a six-bit value
LUN	Logical Unit Number, a two-bit value
Cmd	Command Number
Data1….	Command Data

Cooked IPMI responses have the following format:

<Ccode> <Data1> <Data2> … <DataN>

where

Ccode	Command Completion code
Data1…	Command Response data

4.2.5.2 Setting up the RS485 converter hardware

Any RS485 hardware, such as a RS485 port converter or an RS485 PCI card could be used for the ICMB tests provided they meet the following requirements:

- The RS485 hardware must to present a serial (COMx) interface to system software.
- RS232 serial interface must be able to handle transmission at 19,200bps
- RS485 interface must support 2-wire protocol.

In order to reduce possibility of collisions, ICMB devices arbitrate for the bus. We do not require the RS485 test hardware to implement the arbitration scheme since this may require intimate knowledge of the hardware as well as modifications to the hardware and the OS driver software. The ICMB transport module and tests assume that the ICMB segment consists of only two devices: the target ICMB device and the RS485 hardware.

The ICMB transport module is verified to work on the RS485 port converter, 485OI9TB from B&B electronics (http://www.bb-elec.com). The wiring for the RS485 port converter is as follows:

1. The echo pin should be in the OFF position
2. Tx(A), Rx(A) pins must be wired together and connected to Tx/Rx(-) wire of the ICMB cable
3. Tx(B), Rx(B) pins must be wired together and connected to Tx/Rx(+) wire of the ICMB cable
4. RS485 side requires external power source. Connect the +12V DC and GND wires to appropriately.

4.2.5.3 Test startup sequence

The target system must be discovered before it responds to ICMB requests. The ICMB tests must execute the following sequence of steps before starting the actual test.

1. Find the ICMB address of the target system. This could be accomplished by reading the target ICMB bridge via SMS or by issuing GetICMBAddress ICMB command to broadcast address.
2. Send a SetDiscovered command to the target ICMB bridge device.

4.2.6 The SMB Transport Module

The following Table 4-6 lists the cross-layer communication characteristics and SMBTRANS implementation for the SMB Transport Module:

Table 4-6. SMB Implementation

Transport Characteristic	SMBTRANS Implementation
Software Implementation	Three files: smbtrans.tcl, icadrv.dll and icaio.sys / pci_i2c.sys
Logical Transports	None.
Interface Types	IPMI, CFG
Interfaces (IPMI)	SMB
Interfaces (CFG)	CFG
Interface Nodes	None.
Raw Interfaces	I2C-Fully supported. The interface uses polled mode.
Target Hardware Requirements	None.
Target Software Requirements	None.
Host Hardware Requirements	ICA90 ISA card or PCI90 PCI card (Saelig card) with IPMB cable connected to the target system.
Host Software Requirements	None.
Host Operating System	WinNT, Win2000
Tcl/Tk Version	Tested with 8.3.0 through 8.3.9
`topen` portid	ICA90 card base address (default address is 0x310) or 0-based PCI90 card number.
`topen` options	ICA90 or PCI90 card's I2C slave address. Default slave address is 0x54. May be changed with ConfigTool.
`topen` limitations	One open port at a time.
`tsend/tget` limitations	Message queue handles up to 64(0x3F) messages.
Incoming queue limitations	Message queue handles up to 64(0x3F) messages.
Timeout Implementation	Default value is 1 seconds, but set in I_tranget.tcl to 6 seconds. May be changed with ConfigTool.
Optional Procedures	`terror, tdebug`
Non-standard Procedures	None.

Debug Levels

The debug levels for the SMBTRANS implementation of `tdebug` are as follows:

Level 0: No messages.

Level 1: Print all incoming cooked or raw data based on the type of call.

Level 2: Print all outgoing cooked or raw data based on the type of call, plus level 1 messages.

Level 3: Print all outgoing and incoming, i.e., level 1 messages plus level 2 messages.

4.2.7 CFG Interface

The "CFG" interface is a non-IPMI interface that can be used to send configuration commands to the SIOTRANS, IPMBTRANS, ICMBTRANS and SMBTRANS transports using tsend and tget. A cooked CFG command for tsend has the form:

<Configuration cmd> <data1> <data2> … <dataN>

The response format from tget for the CFG interface is:

<data1> <data2> … <dataN>

The configuration commands and their parameters are shown in Table 4-7.

Table 4-7. CFG Configuration Commands

Command Name	Cmd	Request Data	Response Data
GET_TIMEOUT	0	None	Completion Code, milliseconds
SET_TIMEOUT	1	In milliseconds (1000 is 1 second)	Completion Code
GET_RETRY	2	None	Completion Code, Integer (Default is 3)
SET_RETRY	3	Integer	Completion Code
SET_SLAVE_ADDRESS	4	For ICMB Node Address – LSB, MSB or Hex byte for Slave Address.	Completion Code
GET_SLAVE_ADDRESS	5	None	Completion Code, for ICMB Node Address – LSB, MSB or Hex byte for Slave Address
SET_TARGET_ADDRESS	6	Node Address – LSB, MSB	Completion Code
GET_TARGET_ADDRESS	7	None	Completion Code, Node Address – LSB, MSB

The CFG configuration commands can be done using the Configuration Tool. If the command is not allowed by the transport then a message is printed out about invalid configuration command. The tool can be loaded into ICTS from the tests\cfgtools folder and is called ConfigTool.tcl. The Configuration Tool with the different commands that then tool can perform is shown in Figure 4-2.

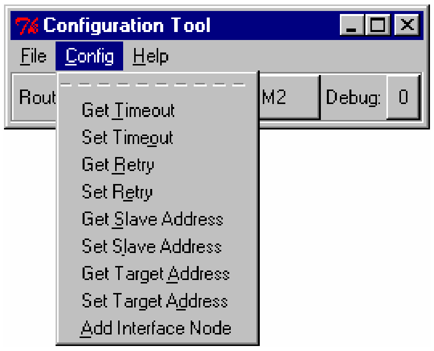


Figure 4-2. Configuration Tool with Configuration Commands

4.2.8 The Serial Transport Module

The following Table 4-8 lists the cross-layer communication and the IOSTRANS implementation for the IOS Transport Module:

Table 4-8. IOSTRANS Implementation

Transport Characteristic	IOSTRANS Implementation
Software Implementation	Single file, iostrans.tcl
Logical Transports	None
Interface Types	IPMI, CFG
Interfaces (IPMI)	IOSTRANS: IOS
Interfaces (CFG)	IOSCFG
Raw Interfaces (IPMI)	Yes, the transport fully supports a raw interface.
Target Hardware Requirements	Available serial connector on host computer. Null modem cable (for direct connect) or modem to modem hardware
Target Software Requirements	None
Host Hardware Requirements	Serial port supporting the basic mode serial interface with either direct connect and/or modem connect modes.
Host Software Requirements	None.
Host Operating System	Win9x, WinNT, Win2000
Tcl/Tk Version	8.3.0 through 8.3.9
`topen` portid	COM port name (COM1, COM2, etc.) No default.
`topen` options	See section 3.2.6.2
`topen` limitations	One open port at a time.
`tsend` limitations	None
`Tget` limitations	None
Incoming queue limitations	None
Timeout Implementation	May be changed with the IOS configuration tool (iosTool.tcl see below).
Optional Procedures	tdebug
Non-standard procedures	None.

NOTE

See Appendix for Null Modem description and layout.

4.2.8.1 Specifications and Characteristics

The IOS transport module allows three levels of debug through `tdebug`. Setting debug to zero blocks displaying of any messages. Levels 1-3 show increasing levels of detail.

IOS Raw Messages

The trawsend/trawget procedures are implemented by the IOS transport. The format of messages is defined by the IPMI specification. The routine calling trawsend is responsible for all packet details including escaping and framing of the packet. Routines receiving data through Trawget must decipher the packet including removing framing and character escaping.

Cooked Message Format

The IOS transport is an IPMI interface and hence the cooked message format of IOS is identical to the IPMI cooked message format.

<dest> <netfn> <lun> <cmd> <data> <…>

where:

dest	Destination micro's I2C Slave Address
netfn	Network function, a six-bit value
lun	Logical Unit Number, a two-bit value
cmd	Command number
data	First command data byte
…	Additional data bytes

Cooked IPMI responses have the following format:

<Ccode> <Data1> <Data2> … <DataN>

where

Ccode	Command Completion code
Data1…	Command Response data

4.2.8.2 IOSTRANS Port Options

The configuration options available on opening the transport include:

- Baud
  This defines the baud rate setting of the testing unit's serial port. If connecting to a BMC via a direct connection, this value must match the BMC's baud rate setting. If connecting via a modem, the value must match the modem's baud rate setting.

- ModemInit
  This is a string used to initialize a host computer's modem if the transport is commanded to dial a connection to the BMC via a modem.

- ModemDial
  This is a string used to define a host computer's modem dial sequence if the transport is commanded to dial a connection to the BMC via a modem.

- PingWaitTime
  This defines how long the transport will wait for a ping before assuming that the BMC is not pinging.

- CommandTimeOut
  This defines the period of time in milliseconds that the transport will wait to receive a response from the BMC when the user requests to receive a response (via tget).

- PacketTimeOut
  This defines the period of time in milliseconds that the transport expects the BMC to finish an individual transmitting an individual packet. I.e. the transport will timeout if the time between a received message's frame start (0xa0) and the message's frame end (0xa5) exceed this value.

- ACKCheck
  This (value [0,1]) tells the transport whether to check for packet acknowledgments after sending a packet.

- WaitForPing
  This (value [0,1]) tells the transport whether to check for a ping message prior to sending a packet.

- RetryCount
  Specifies the number of times a packet is resent if the response is not received by the BMC. Note, this parameter's functionality is not implemented at this time.

- SequenceNumber
  This parameter allows one to configure the first sequence number to be used by the transport. Note, this number can range between 0 & 0x3f and is incremented and possibly wrapped to '0' prior to use.

These configuration options are sent to the transport upon opening the transport. They are defined in the Target_PortOptions(IOSTRANS) variable in the target configuration file. The parameters are defined in a string of the form "var1=val1 var2=val1…." The port options variable can be undefined or defined to an empty string if no options are desired, or can have as many pairs as the users wishes. The variable definition pairs are separated by spaces. There should not be any spaces surrounding the equals sign. All of these parameters can be changed after the transport has been opened. See the next section for details about transport parameters and utility functions.

4.2.8.3 IOSTrans configuration options

The transport supports a rich list of configuration options available through tsend's IOSCFG interface. The options include:

isBMCPinging
This command will return data indicating whether a ping has been received and what type of ping (session active, session not active, mux switch set to system). Optionally, one may pass a string parameter with the value "noflush" to indicate that the routine should not flush the pings prior to checking for a ping.

isModemConnected
The command returns a true/false indication indicating whether the transport has an active connection through a modem.

dialModem
Executing this command causes the transport to dial a phone number using a modem. This command takes an optional string parameter defining the phone number to dial. If the optional parameter is not defined, the default string used is defined by the port options variable "ModemDial". This default can be manipulated with the setPhoneNum and getPhoneNum functions described below.

hangupModem
As one might guess, this causes the transport to hang up an existing modem connection.

muxBMC
This command causes the tranport to send two bytes (0x1B 0x28, "esc-(") out the serial port to cause the system to switch the mux to the BMC.

muxSystem
This command causes the tranport to send two bytes (0x1B 0x51, "esc-Q") out the serial port to cause the system to switch the mux to the system.

getModemState
This command returns an indication of the modem connection status.

closeAndOpenNewConnection
The command causes the tranport to close an re-open the serial port.

setRetryCount
This command is used to specify the number of retries the transport will attempt if a firmware command is sent, but a reply is not received. Use of the "retry count" variable is not implemented, so at this time, this command does not impact the transport's operation.

getRetryCount
This command is used to retrieve the number of retries the transport will attempt if a firmware command is sent, but a reply is not received. Use of the "retry count" variable is not implemented, so at this time, this command does not impact the transport's operation.

setBaudRate
This command takes a single parameter defining the new baud rate to use.

getBaudRate
The command returns the current baud rate.

setPhoneNum
This command defines the default phone number to call when making a connection through a modem.

getPhoneNum
The command causes the transport to return a sting indicating the current default phone number to use when connecting through a modem.

setAckChecking
This command take a true/false (1/0) parameter indicating whether the transport should check for BMC acknowledgements when sending messages to the BMC.

getAckChecking
This command returns the current state of BMC acknowledgment checking.

setPingChecking
This command takes a true/false (1/0) parameter indicating whether the transport should verify that the BMC is pinging before sending a message to the BMC.

getPingChecking
This command returns the current state of the transport's "ping checking".

getAckCount
Returns the number of BMC acknowledgements received since the transport was started.

getPortHandle
Return's the file handle to the serial port used by the transport.

setDebugLevel
This command accepts a single integer argument ranging from 0-3. The variable is used to increase or decrease the amount of debug information printed by the transport. The value '0' causes almost no information to be printed during the operation of the transport. The value '3' causes more information then most people can stand to be printed during the operation of the transport.

getDebugLevel
Returns the transport's current debug level setting.

setPingTimeout
The command expects a single integer parameter defining the maximum amount of time in milliseconds to wait for a ping message from the BMC. Note: if this value is shorter than the BMC's ping interval, then the transport will not be able to correctly determine if the BMC is pinging.

getPingTimeout
Returns the ping timeout.

setCommandTimeout
The command expects a single integer parameter defining the maximum amount of time in milliseconds to wait for a command response message from the BMC.

getCommandTimeout
Returns the command timeout.

setSequenceNumber
Defines the next sequence number for the transport to use. Note, because of the design of the transport the actual sequence number used will be 1 greater than the value specified by this command. Sequence numbers are incremented until the value 0x3f is reached, at which time the sequence number is reset to '0'. If a sequence number <1 or greater than 0x3f is sent, the next sequence number used will be '0'.

getSequenceNumber
Returns the current sequence number.

To send a configuration command to the transport, one should use the following examples:

```
puts [msendget IOSCFG [list setCommandTimeout 2000]]

puts [msendget IOSCFG [list getCommandTimeout]]
```

A graphical menu based tool has been written to expose all of these configuration commands. The tool is called iostool.tcl and is show in Figure 4-3. IOS Tool.

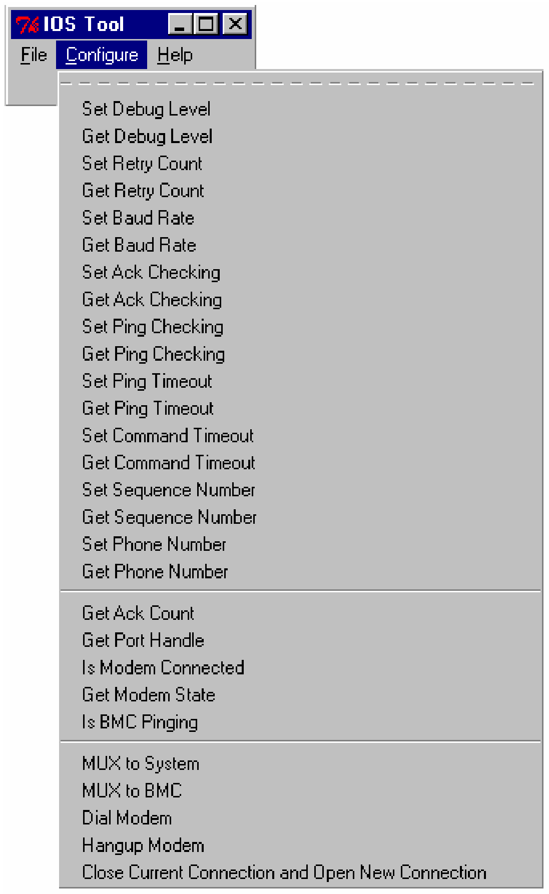


Figure 4-3. IOS Tool

4.2.9 The LAN Transport Module

The following Table 4-9 lists the cross-layer communication characteristics and IOLTRANS implementation for the IOL LAN Transport Module:

Table 4-9. IOLTRANS Implementation

Transport Characteristic	**IOLTRANS Implementation**
Software Implementation	ioltrans.tcl, xtrans.dll, IOLTRANS.dll
Logical Transports	None
Interface Types	IPMI, IOLCTRL
Interfaces (IPMI)	IPMI Over LAN
Interface Nodes	None.
Raw Interfaces	None
Target Hardware Requirements	Firmware supporting IPMI 1.5 LAN channel.
Target Software Requirements	None
Host Hardware Requirements	Ethernet adaptor connected to a common network with the target.
Host Software Requirements	WinNT, Win2000
Host Operating System	Windows
Tcl/Tk Version	Tested with 8.3.0 through 8.3.9
Topen portid	An IP address in dot notation. For instance, 222.222.222.99
Topen options	A list of strings of the form, -*option* <*value*>. The option names can be entered as non-ambiguous partial strings (see table below).
Topen limitations	Currently limited to one session.
Tsend/tget limitations	Currently limited to one message transaction at a time.
Incoming queue limitations	Currently one.
Timeout Implementation	One second default. Timeout is specified as milliseconds. Timeout is on a per-message basis and defines how long the transport will wait for a response before retrying the same packet.
Optional Procedures	terror, tdebug
Non-standard Procedures	Transport provides an API for use in test scripts. The commands are defined in the following sections.

4.2.9.1 IOLCTRL Interface

The IOLCTRL interface supports non-numeric, string based, configuration parameters. Configuration commands are sent to the transport via this interface using tsend:

```
tsend <pd> IOLCTRL <-config_opt> <value> … <-config_opt> <value>
```

pd is the descriptor that was returned by topen and represents an instance of an IOLTRANS session object.

In a similar manner, configuration options can be passed to the transport when opening a new connection with the topen command:

```
topen <-config_opt> <value> … <-config_opt> <value>
```

`tsend and topen` are internal commands used by the framework. Test script developers will most likely not access these direction. Direct access to the configuration options below is provide through the API command, `iol_tconfigure`. This command is documented in the next section.

There are two components to IOLTRANS, one to handle IPMI 1.5 sessions and the other handles IPMI 2.0 (RMCP+) sessions. They both share many of the same configuration options; however, there are some differences as illustrated in the "Mode" column of the following table.

The Table 4-10 represents the configuration options and their required data.

Table 4-10. Configuration Options

Configuration Option	Request Data	Mode	Notes
-protoVer	[1.5|2.0]	1.5/2.0	This option is only writable via topen. This is indirectly set using the startup batch file and defining IOLOPT. Otherwise, it is read-only. When set to 1.5, the transport is operating per the IPMI 1.5 specification. When set to 2.0, it is communicating according to RMCP+ specifications.
-password	<password string>	1.5/2.0	The password the transport will use to calculate the authentication code in the session header. This option will zero extend or truncate the password to 16 bytes if needed.
-username	<user name string>	1.5/2.0	The username the transport will use to connect to the target.
-authtype	[none|straightpw|md2|md5]	1.5	The authentication type to use to establish the session. One of the strings listed to the left must be passed to this option.
-rakpAuth	[none|hmac_sha1|hmac_md5]	2.0	Sets the authentication algorithm used for RMCP+ session establishment.
-privlev	[callback|user|operator|admin]	1.5/2.0	The maximum Privilege Level the transport will request when connecting.
-managesession	[0|1]	1.5	Turns on/off the thread in the transport that occasionally pings the target and automatically reestablishes sessions if lost.
-timeout	<timeout value in ms>	1.5/2.0	Defines how long the transport will wait for a response for a given packet before resending it.

-retries	<number of retries>	1.5/2.0	The number of times to retry a packet after the initial packet doesn't receive a response.
-sessionSeqNum	<new sequence number>	1.5/2.0	This is the "inbound" sequence number from the BMC's perspective. Typically, this value should not be set unless testing this value is being done.
-sessionId	<new session ID to use>	1.5/2.0	This forces the transport to update the session ID in the session header without performing any additional steps. This value is also not typically changed, and the transport handles this value automatically.
-reconnect	[0|1]	1.5/2.0	Determines if the transport will assume a session is lost when a response is not received and attempt to connect a new session. Default is set to 0 (off).
-debug	<value 0+>	1.5	If set to >0, this causes the transport to send error, warning, and status messages to the FTF main window.
-payloadType	[ipmi_msg|sol|oem|open_sess_req|open_sess_rsp|rakp_1|rakp_2|rakp_3|rakp_4]	2.0	Sets the payload type field value in the RMCP+ session header.
-portNum	<integer>	1.5/2.0	Sets the management port value. Defaults to 623 in accordance with the IPMI specifications.
-sessActive	Read only [1|0]	1.5/2.0	Reports whether a session is currently active or not.
-sessMgrCount	Read only integer	1.5/2.0	Reports the current number of session manager objects instantiated in the transport.
-encryption	[none|aes_cbc_128]	2.0	Selects the RMCP+ encryption algorithm to use for the session.
-incrInbSeqNum	<integer>	1.5/2.0	Changes the sequence number for packets sent to the BMC. Used for testing only.
-integrity	[none|hmac_sha1|hmac_md5|md5]	2.0	Selects the RMCP+ integrity algorithm.
–lookupMode	[user_priv|name_only]	2.0	User_priv means both the request username and privilege will be used to lookup password key. This option value can be used combine with null username to enable "role only" login Nam_ only means user name alone is used to look up password key. Privilege level option is acts as a 'Maximum request privilege level' as in IPMI1.5

4.2.9.2 IOLTRANS Library Commands

In addition to supporting the IOLCTRL interface, the IOL transport implements several useful library commands that give the user a direct interface to these same features. These commands can be useful in scripting tests, tools, or in constructing a more versatile library. These commands are registered in the global namespace when the transport is loaded.

```
iol_tconfigure <pd> <config_opt> <data> … <config_opt> <data>
```

This command is constructed from the same code used to implement the IOLCTRL interface and accepts the same configuration options and values.

```
iol_tconnect <pd>
```

Causes the transport to connect the session object referred to by the descriptor to the target. In other words, the transport will attempt to establish a session using the specified session object.

```
iol_tdisconnect <pd>
```

When issued, the session object, pd, will send a Close Session request and terminate the session.

```
iol_terror <ecode>
```

Provides access to the ecode translator for the transport. Will return a short string describing the error value.

```
iol_tgetDescriptorList
```

Returns a Tcl list of descriptors. These descriptors can be used as parameters to the procedures described in this section.

In addition to supporting the standard test API for IPMI 1.5, IOLTRANS, when operating in IPMI 2.0 RMCP+ mode, provides the following additional commands for use in scripting. The following commands do not apply to IOLTRANS when operating in RMCP mode per the IPMI 1.5 specification.

```
iol_tcreate
```

This procedure creates a session manager, or connector object, inside the transport and returns a descriptor for the object. The new instance will need to be configured with an IP before rmcpp_tconnect is issued.

```
iol_tdelete <pd>
```

This command deletes the session manager object associated with the given descriptor.

```
iol_warnAsFail <pd>
```

This procedure configures the transport to send messages normally displayed as warnings to the ICTS window as fail messages.

4.2.9.3 Debug Levels

Currently, IOLTRANS supports only one tdebug level. When –debug is set to any value equal to or greater than 1, the transport will send messages to the main FTF window. These messages include error, warning, and status messages helpful in determining any problem the transport may encounter with a session.

4.3 Installation

The ICTS system includes components from several sources. This section contains instructions for acquiring the components necessary to install, configure, and run ICTS. Before attempting to install the system, refer to the *Hardware Requirement* and *Software Requirement* sections of this chapter to be sure your host system can support ICTS.

4.3.1 Downloading Tcl/Tk

Before installing ICTS, you must have Tcl/Tk version 8.3.0 through 8.3.9. If you do not already have Tcl/Tk, complete the following procedure:

1. Log into the Web site: *http://www.scriptics.com/software/8.3.html*.
2. Scroll down to the section titled "Downloading Binary Releases for Windows and Macintosh" and download the self-installing executable file for Windows 95, Windows 98, or Windows NT/2000.
3. Run the downloaded file on the host machine. Run any setup and configuration required in the readme file.

4.3.2 Downloading the Test Framework Suite

Once Tcl/Tk has been installed on the host system, you will need to download a copy of the test framework suite. To prepare for framework installation and to download the framework software, complete the following procedure:

1. Back up your existing framework installation.

⚠ **CAUTION**

Installing a new version of the framework over an old version will overwrite any files that you have customized. Creating a backup copy with WinZip† or another archiving utility will allow you to retrieve these customized files.

2. Log in to the Web site: http://developer.intel.com/design/servers/ipmi/
3. Download the self-extracting executable file containing the latest version of the framework.
4. Run the downloaded file in a temporary directory. Installation scripts, an archive file, and a text file named README.TXT appears in your temporary directory.
5. Read the README.TXT file.

4.3.3 Installing Firmware Test Framework and Configuration Files

Installation of the Firmware Test Framework requires the hardware and software described in the *ICTS Operating Requirements* section of this chapter. Before attempting to install the Firmware Test Framework, be sure the hardware and software meet the requirements.

Unless directed to do otherwise by the `README.TXT` file extracted from the downloaded archive file, install the firmware test framework on your PC using the following procedure:

1. Execute the extracted file `setup.tcl` using one of the following two methods:
   - *Click* (or double-click) on file `setup.tcl` from the Windows File Manager or Windows Explorer†.
   - Enter the following command at a DOS command prompt:

     ```
     wish83 setup.tcl
     ```
2. Answer the questions asked by the setup program.
3. Shutdown and reboot the system.

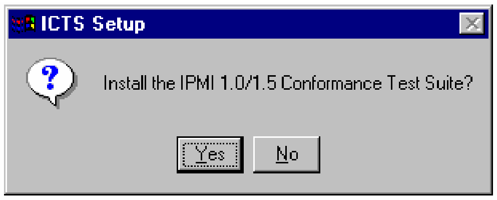
   If you later want to uninstall the software this may be done with the following DOS command:

```
wish83 setup.tcl uninstall
```

The prompts in the installation process are next shown.

4.3.3.1 Installation Prompt

The installation prompt allows installation to continue or not to continue.



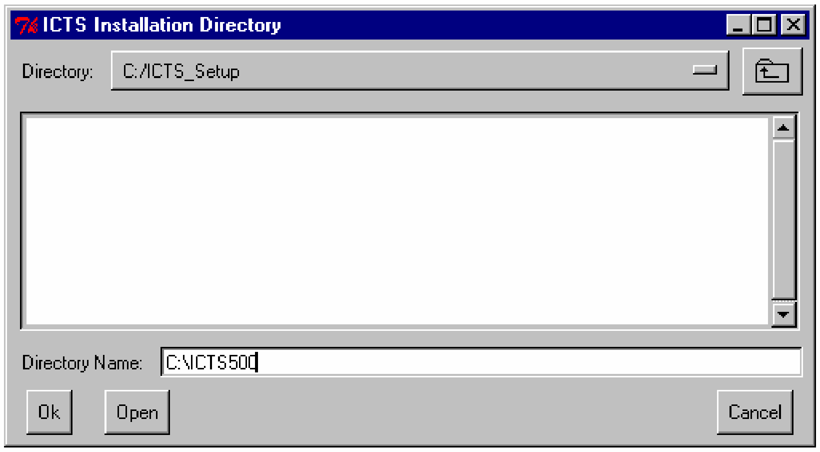
ICTS Setup

Install the IPMI 1.0/1.5 Conformance Test Suite?

Figure 4-4. Installation Prompt

4.3.3.2 Installation Directory

Next the installation directory prompt is shown in **Error! Reference source not found.**. Types in a directory name to have the installation create a new directory.



ICTS Installation Directory

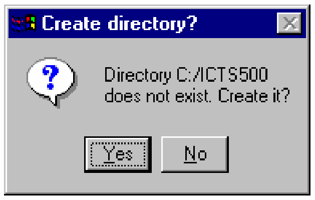
Directory: C:/ICTS_Setup

Directory Name: C:\ICTS500

Figure 4-5. Choosing Installation Directory

4.3.3.3 Create Directory

The create directory prompt is shown in **Error! Reference source not found.**. Deciding not to create the directory will display the installation directory prompt again.



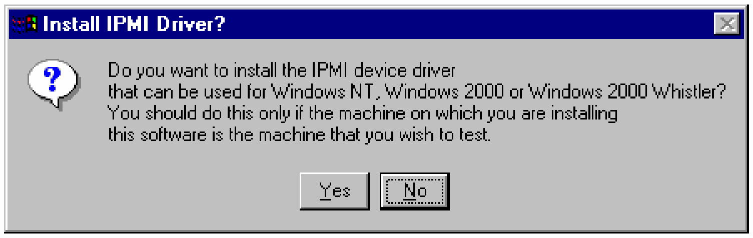
Create directory?

Directory C:/ICTS500
does not exist. Create it?

Figure 4-6. Create Directory

4.3.3.4 Install IPMI Driver

The install IPMI Driver prompt is shown in **Error! Reference source not found.**. The IPMI Driver will enable ICTS to run on the target machine with out needing a host machine. If this is installed on a non IPMI platform, it may result in malfunctioning of the system. Eg. Sytem might not boot up after installation.

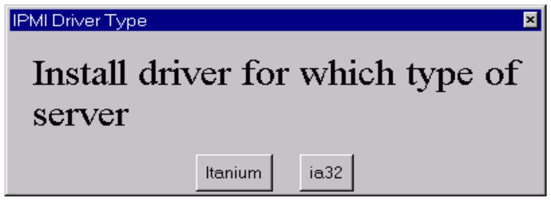

Install IPMI Driver?

Do you want to install the IPMI device driver
that can be used for Windows NT, Windows 2000 or Windows 2000 Whistler?
You should do this only if the machine on which you are installing
this software is the machine that you wish to test.

Figure 4-7. Install IPMI Driver

4.3.3.5 IPMI Driver Type

The IPMI driver type prompt is shown in **Error! Reference source not found.**. Select the IPMI Driver type of server.


IPMI Driver Type

Install driver for which type of server

Figure 4-8. Install Driver Type

4.3.3.6 Install ICA90 and PCI90 Driver

The ICA90 driver type prompt is shown in **Error! Reference source not found.**, and the ICA90 driver type prompt is shown in **Error! Reference source not found.**. The drivers are for the ICA90 and PCI90 cards.

WARNING

Do not install the driver that came with the card. Otherwise the ICTS driver will not work. See note below on how to remove the driver.

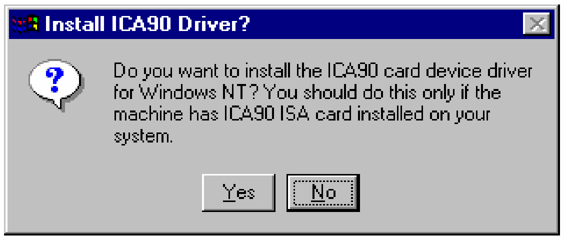
NOTE

For Windows 2000:

Start / Control Panel / Add/Remove Hardware / Uninstall / Unplug a device / Uninstall a device / check show hidden devices / select "Isa_i2c or Pci_i2c", Ok? / Check, Yes I want to uninstall this device.

Windows NT:

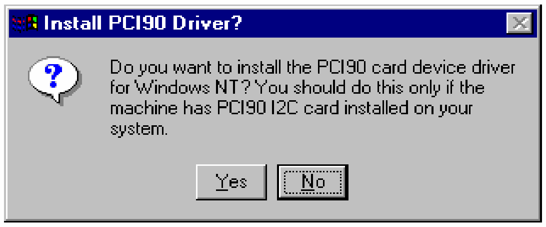
Start / Control Panel / Devices / select “Isa_i2c or Pci_i2c”, then select stop.



Install ICA90 Driver?

Do you want to install the ICA90 card device driver for Windows NT? You should do this only if the machine has ICA90 ISA card installed on your system.

Figure 4-9. Install ICA90 Driver



Install PCI90 Driver?

Do you want to install the PCI90 card device driver for Windows NT? You should do this only if the machine has PCI90 I2C card installed on your system.

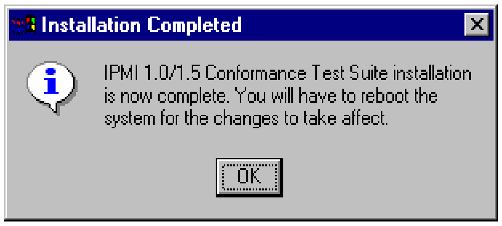
Figure 4-10. Install PCI90 Driver

NOTE

There is not an INF file for the ICA90 or PCI90 drivers so Windows 2000 will complain, but the drivers will still work.

4.3.3.7 Installation Completed

The installation completed prompt is shown in **Error! Reference source not found.**. Now that the installation is complete ICTS now needs to be comfigured. This will be explained in next.



Installation Completed

IPMI 1.0/1.5 Conformance Test Suite installation is now complete. You will have to reboot the system for the changes to take affect.

Figure 4-11. Installation Complete Prompt

4.4 Configuration

Before attempting to use the Firmware Test Framework, environment variables must be set or confirmed, directories must be organized according to the structure described in this chapter, and the host, platform, and target must be configured to allow communication between them. This section contains procedures for configuring your Firmware Test Framework.

4.4.1 Directory Layout

The installation procedure creates a number of directories. Some must appear as the installation process created them. Others may be renamed or moved. The following table lists the fixed directories and a description of their contents:

Table 4-11. Fixed Directories

Name	Description
base\	Tcl files for the base framework
bin\	Batch files for starting the framework
dll\	Dynamic load libraries
docs\	Documentation
hosts\	Default directory for host configuration files
libs\	Tcl files for framework libraries
local\	Local configuration files – Always starts out empty, immune to being overwritten by future upgrades
Packages\	Tcl files for framework packages
Templates\	Templates and examples for configuration files and test modules

In addition to the fixed directories, a number of directories allow user manipulation. The following table lists the directories that can be moved or renamed:

Table 4-12. Configurable Directories

Name	Description
logs\	Default directory for log files.
Platforms\	Default directory for platform configuration files and command matrix files.
setup\	Default directory for test setup files.
targets\	Default directory for target configuration files.
tests\	Default directory for test modules and tool modules.
tmp\	Default directory for scratch files.
Transports\	Default directory for transport modules.
users\	Default directory for user configuration files.

After installation is complete and you have arranged the user-definable directories on your host system, you must create configuration files for the host, the platform, and the target. You may also want to customize batch files or create a configuration file for each user. This section contains procedures for configuring ICTS, customizing batch files, and creating user configuration files.

4.4.2 Configuration Parameters

After installing ICTS, ICTS needs to be configured in order use the different transports. The following Table 4-13 shows configuration parameters. The config.txt file in the Docs\ directory also describes the configuration parameters.

Table 4-13. Configuration Parameters

Variable	Options	Set Value
ICTSROOT		Your ICTS directory name. Please use an 8.3 filename format (xxxxxxxx.yyy) instead of a long file/directory name.
ICTSSHELL		Your Tcl/Tk interpret name such as wish83.exe.
IPMI	1.00 or 1.50 or 2.00	IPMI version, the default version is 1.00.
ICTSMODE	NORMAL (default) DEVELOPER	The ICTS mode you want to use. If set to NORMAL, the ICTS runs in the NORMAL mode, and the user cannot develop a self-test case. If set to DEVELOPER, the ICTS runs in the DEVELOPER mode, and the user can run the ICTS test case and also develop a self-test case.
ICTSTRANS	SIOTRANS LOCTRANS (default) NONE	The transport you want to use for IPMI testing using the KCS/SMIC interface or SMS interface. If you don't specify an option, the default value is LOCTRANS. To disable ICTSTRANS, set it to NONE. LOCTRANS works only on Windows NT† 4.0.
ICTSCOM	COM1 COM2 COM3 COM4	The serial port used in the SIOTRANS mode. If you select LOCTRANS, this variable will be ignored.
IPMBTRANS	ICATRANS	The transport you want to use for IPMB testing using the I2C interface. To disable the IPMB interface, unset the IPMBTRANS environment variable by executing: set IPMBTRANS=NONE
IPMBPORT	0x310 0 1	ISA default port etc. Port First PCI card determined by the system. Port Second PCI card determined by the system.
ICMBTRANS	ICMBTRANS	The transport you want to use for ICMB testing. To disable the ICMB interface, unset the ICMBTRANS environment variable by executing: set ICMBTRANS=NONE
ICMBCOM	COM1 COM2 COM3 COM4	The serial port to be used in the ICMBTRANS mode.
SMBTRANS	SMBTRANS	The transport you want to use for SMBus testing using the I2C interface. To disable the SMBus interface, unset the SMBTRANS environment variable by executing: set SMBTRANS=NONE

SMBPORT	0x310 0 1	ISA default port etc. Port First PCI card determined by the system. Port Second PCI card determined by the system.
IOSTRANS	IOSTRANS	The transport you want to use for Serial/Modem testing. To disable the IOS interface, unset the IOSTRANS environment variable by executing: set IOSTRANS=NONE
IOSCOM	COM1 COM2 COM3 COM4	The serial port to be used in the IOSTRANS mode.
IOLTRANS	IOLTRANS	The transport you want to use for LAN testing. To disable the LAN interface, unset the IOLTRANS environment variable by executing: set IOLTRANS=NONE
IOLPORT	XXX.XXX.XXX.XXX	The IP address of the target.
IOLOPT	<-option> <value>...	The IOLTRANS options. These options can be catenated in one string to set the IOLOPT variable. The LAN transport implements partial word completion on the parameters, so a sub-string of the option name or value can be provided if it is non-ambiguous. For instance "-priv admin" would be translated as "-privlev administrator".
	-privlev [callback|user|operator|administrator]	This option sets the privilege level that the transport will attempt to establish a session at.
	-authtype [none|md2|md5|straightpw]	This option sets the authentication type that the transport will use.
	-timeout <value in milliseconds>	This option determines how long the transport will wait for a response before resending the packet.
	-retries <value>	This option sets the number of times to retry a packet if no response is received.
	-debug [0|1]	This option turns off/on transport messages that appear in the FTF main window. These messages include error, warning, and status messages. This is off by default.
	-manageSession[0|1]	This option determines whether the transport will periodically send a packet to the target to keep the session alive. This is on by default.
	-reconnect [0|1]	This option determines if the transport will attempt to establish a new session if a response is not received to a packet after all retries have been

	-password <string> -username <string> -protoVer [1.5|2.0]	completed. The default is on. This option sets the password string that the transport will use to connect with. It is null by default. This option sets the user name that the transport will use to connect with. It is null by default. This option sets the transport in RMCP+ (IPMI 2.0) or RMCP (IPMI 1.5) session protocol mode. An example string of options and values follows, the user may use their own discretion in these settings: IOLOPT=-priv admin -auth md2 -timeout 3000 -retries 2 -debug 1 -manage 1 -reconnect 0
ADDMICRO	micro-controller micro-controller	Used to setup additional micro-controllers other than BMC. set ADDMICRO= micro-controller micro-controller etc.
MFGTESTONPASS	micro-controller password micro-controller password	Used to setup manufacture password on micro-controllers set MFGTESTONPASS=micro-controller password micro-controller password etc.

Examples: set ICTSROOT=C:\ICTS

set ICTSSHELL=C:\TCL\bin\wish83.exe

set ICTSMODE=DEVELOPER

set ICTSTRANS=SIOTRANS

set ICTSCOM=COM1

set IPMBTRANS=ICATRANS

set ICMBTRANS=ICMBTRANS

set ICMBCOM=COM2

set ADDMICRO= micro-controller micro-controller etc.

set MFGTESTONPASS=micro-controller password micro-controller password etc.

4.4.3 The Batch File

In directory `C:\FTF\bin`, the files `ICTS.BAT` (for normal mode), `ICTSDEV.BAT` (for developer mode), or `ICTSDEV5.BAT` (for developer mode IPMI 1.5) are batch files that start the framework in an ICTS configuration. If `ICTSROOT` is undefined, it attempts to define the root for itself. It also makes a guess at the name of the Tcl/Tk interpreter and sets the `ICTSSHELL` environment variable accordingly. Make sure the path is set to C:\Program Files\tcl\bin default or wherever Tcl/Tk is installed. To stop it from guessing, hardcode the values for these variables at the top of the batch file or elsewhere before running the batch file. (If you used the `setup.tcl` script for performing the ICTS installation, the `ICTSROOT` and `ICTSSHELL` variables are defined automatically.)

The other configuration parameters can be put into the environmental variables, put in `ICTS.BAT, ICTSDEV.BAT or ICTSDEV5.BAT`, or put into a separate batch file that is run before staring ICTS. To start ICTS use `ICTS.BAT, ICTSDEV.BAT, or ICTSDEV5.BAT` by running it from the command prompt. `ICTS.BAT, ICTSDEV.BAT, or ICTSDEV5.BAT` starts the FTF with the ICTS default host and user configuration files described in the fallowing sections. The user configuration file in turn causes the default platform and target configuration files to be loaded along with the ICTS test modules.

4.5 Customizing the Batch File

Custom versions of `ICTS.BAT` could reference alternate configuration files. Normally, you will not need custom configuration unless you are developing new software modules for ICTS. If creating the necessary additional configuration files, you can create additional batch files that automatically load the framework with specific combinations of configuration files containing parameters described in the *Host Configuration Files* section of this manual. For additional information about using the developer's mode, refer to the *Intelligent Platform Management Interface (IPMI) Conformance Test Suite (ICTS) Developer's Guide*.

4.5.1 Host Configuration Files

The host computer is where ICTS runs and where you will work. Before loading any tests, you must create a host configuration file and place it in the directory `C:\FTF\hosts` (assuming `C:\FTF` is your FTF installation directory) In this directory you can find the default host configuration file for ICTS, `i_host.tcl`. Normally, you will not have to modify this file unless you are developing new software modules for use with ICTS.

In addition to the generic configuration files, the directory `C:\FTF\templates\config` contains configuration file templates, including `hostcfg.tcl`. The templates are commented to show required and optional variables and what procedures may be defined in a host configuration file.

4.5.1.1 Host Configuration Variables

The following table lists the configuration variables from the `hostcfg.tcl` file. Each variable is accompanied by a description of its use, an example, and whether it is required:

Table 4-14. hostcfg.tcl Variables

Variable Name	Purpose	Required	Usage
Host_Name	Identifies host computer	No	set Host_Name "Joe User's Desk"
Host_Description	More verbose identification of the host computer	No	set Host_Description "PC on Joe's Desk, runs NT 4.0"
Host_Revision	Revision number for this configuration file	No	set Host_Revision "0.83"
Host_Transports	List of transport modules to load, in priority order. Do not include logical transports	Yes	set Host_Transports [list loctrans fwhtrans]
Host_Ports	Array of lists indicating valid port IDs for the various transport modules, including logical transports	No	set Host_Ports(FWHTRANS) [list "com1" "com2" "com3" "com4"] set set Host_Ports(FWH-I2C) [list "com1" "com2" "com3" "com4"]
Host_TransDirs	List of directories in which to find transport modules	No	set Host_TransDirs [list "c:\ftf\transports"]
Host_PlatDirs	List of directories in which to find platform configuration files	No	set Host_PlatDirs [list "c:\ftf\platforms"]
Host_TargDirs	List of directories in which to find target configuration files	No	set Host_TargDirs [list "c:\ftf\targets"]
Host_SetupDirs	List of directories in which to find test setup files	No	set Host_SetupDirs [list "c:\ftf\setup"]
Host_TestDirs	List of directories in which to find test modules	No	set Host_TestDirs [list "c:\ftf\tests"]
Host_UserDirs	List of directories in which to find/save user configuration files	No	set Host_UserDirs [list "c:\ftf\users"]
Host_Dirs	Default for the above directory lists	No	set Host_Dirs [list "c:\ftf"]
Host_LogDir	Directory for writing log files	No	set Host_LogDir "c:\ftf\logs"
Host_LogFile	Name for the log file	No	set Host_LogFile "joeuser.log"

Continued

Table 4-14. hostcfg.tcl Variables (continued)

Variable Name	Purpose	Required	Usage
Host_TmpDi	Directory for writing temporary files	No	set Host_TmpDir "c:\temp"
Host_MailHost	Host name (or IP address) of an SMTP server	No	set Host_MailHost mail.mycompany.com

4.5.1.2 Host Configuration Procedures

In addition to the variables listed in the above table, the example host configuration file contains an example of a procedure definition for providing help for information encoded in the configuration variables. Once the procedure is defined, it is available for use by the framework and the framework batch files. The following code listing is a procedure definition for `host_help`. This procedure allows ICTS to print the host name and description to the screen:

```
proc host_help { } {

    global Host_Name Host_Description

    ftf_msg "Name:        $Host_Name"

    ftf_msg "Description: $Host_Description"

}
```

4.5.2 Platform Configuration

Platform configuration files normally contain generic information about the system under test. The configuration file makes that information available to the framework. For example, a Model 1000 platform configuration file provides the framework with information applicable to all Model 1000 systems. Therefore, it is often not necessary to create new platform configuration files.

However, information specific to a particular system under test may be placed in the platform configuration file. The result is that the framework assumes the defined characteristics of the platform are defaults for the target configuration file. It is possible to eliminate the need for a target configuration file by defining a sufficient number of default values in the platform configuration file.

Included with ICTS is a default platform configuration file named `i_platfm.tcl`. This is suitable for most generic IPMI 1.0/1.5/2.0 platforms.

Place platform configuration files in the directory `C:\FTF\platforms` or any directory assigned to the `Host_PlatDirs` variable in your host configuration file.

If you cannot find a platform configuration file to fit your needs, you can create one by copying and modifying the following template file:

```
C:\FTF\templates\config\platcfg.tcl
```

The template configuration file is commented to allow ease of use and clear understanding of all variables and procedures in a platform configuration file. Some variables and procedures are required. Others are optional.

Items unique to the platform configuration file appear in the top half of the template, up to and including the `platform_help` procedure. Target configuration file defaults appear in the lower half of the file.

4.5.2.1 Platform Configuration Variables

The following table lists the configuration variables from the `platcfg.tcl` file. Each variable in the table is accompanied by a description of its use, an example, and whether it is required:

Table 4-15. Platform Configuration Variables

Variable Name	Purpose	Required	Usage
Platform_Name	Identifies target platform (SC450NX, L440GX+, and so forth)	No	set Platform_Name "Model 1000"
Platform_Description	More verbose identification of the platform	No	set Platform_Description "Model 1000 – First IPMI 1.0/1.5/2.0 Platform"
Platform_Revision	Revision number for this file	No	set Platform_Revision "0.80"
Platform_IPMI_Ver	IPMI version number	Yes	set Platform_IPMI_Ver "1.0"
Platform_Possible_uC	List of possible microcontrollers that could be on this platform	No	set Platform_Possible_uC [list BMC]

4.5.2.2 Platform Configuration Procedures

In addition to the variables listed in the above table, the example platform configuration file contains an example of a procedure definition for providing help information encoded in the configuration variables. Once the procedure is defined, it is available for use by the framework and the framework batch files.

The following code listing is a procedure definition for `platform_help`. This procedure allows ICTS to print the platform name and description to the screen:

```
proc platform_help { } {

    global Platform_Name Platform_Description

    ftf_msg "Name:        $Platform_Name"

    ftf_msg "Description: $Platform_Description"

}
```

4.5.2.3 Target Configuration Definitions

This section describes the platform configuration file variables that allow you to define target characteristics. When the framework attempts to resolve relative paths, it tries the path stored in the target configuration file `Host_PlatDirs` variable first. If `Host_PlatDirs` is undefined, then the framework looks for the platform configuration file variable `Host_TargDirs`.

The following table lists the variables that apply to the target along with their purpose and usage:

Table 4-16. Platform Configuration Target Variables

Variable Name	Purpose	Required	Usage
Platform_[other]	Default values for Target variables.	No	set Platform_FW_Ver(BMC) "0.50" set Platform_LogFile "Model1000.log"

4.5.3 Target Configuration

Target configuration files give the framework-specific test system information not captured in the platform configuration file. Most target configuration file settings may have default settings assigned in a platform configuration file. If all default values appear in the platform file and they are appropriate to the specific system under test, the target file is unnecessary.

Included with ICTS is a default target configuration file named `i_target.tcl` for generic IPMI 1.0/1.5/2.0 systems. This will be suitable for ordinary use of ICTS.

If the defaults in the target file are not appropriate for the system under test, create a target configuration file by copying the following template:

```
C:\FTF\templates\config\targcfg.tcl
```

The template configuration file is commented to allow ease of use and clear understanding of all variables and procedures in a target configuration file. Some variables and procedures are required. Others are optional.

Place target configuration files in the directory `C:\FTF\targets` or any directory assigned to the `Host_TargDirs` variable in your host configuration file.

4.5.3.1 Target Configuration Variables

Table 4-17 lists the configuration variables from the `targcfg.tcl` file. Each variable is accompanied by a description of its use, whether it is required, its default value, and an example:

Table 4-17. Target Configuration Variables

Variable Name	Purpose	Req.	Usage
Target_Name	Identifies target box.	No	set Target_Name "targcfg-template" **Default**: The name of the target configuration file.
Target_Description	More verbose identification of the target	No	set Target_Description "Template for target configuration files." **Default**: Target_name
Target_Revision	Revision number for this file.	No	set Target_Revision "0.80" **Default**: 0.0
Target_FW_Ver	Firmware version.	No	set Target_FW_Ver(BMC) "0.50" **Default**: Platform_FW_Ver, if defined.

continued

Table 4-17. Target Configuration Variables (continued)

Variable Name	Purpose	Req.	Usage
Target_LogFile	Log file name.	No	set Target_LogFile "m1000targ.log" **Default**: Platform_LogFile
Target_Interfaces	List of interfaces available on the target.	Yes	set Target_Interfaces [list SMS I2C] **Default**: Platform_Interfaces
Target_SDR_Source	Determines source of SDR information.	No	set Target_SDR_Source Target_Config **Default**: BMC_SDR
Target_Ports	Array of ports you want to open for each transport.	Yes	set Target_Ports(FWHTRANS) "com1" set Target_Ports(FWH-I2C) "com2" **Default**: Platform_Ports, if defined
Target_PortOptions	Array of port open options.	No	set Target_PortOptions(FWHTRANS)"115200" set Target_PortOptions(FWH-I2C) "115200" **Default**: Platform_PortOptions
Target_PortTimeouts	Array of port timeout settings in milliseconds.	No	# Timeout value for the FWH transport set Target_PortTimeouts(FWHTRANS) 10000 **Default**: Platform_PortTimeouts
Target_BadRoutes	List of routes that are unavailable.	No	# Null-modem cable not hooked up on target. set Target_BadRoutes [list "i2c\fwh-i2c"] **Default**: Platform_BadRoutes
Target_IPAddr Target_BroadcastIPAddr Target_IEEEAddr	Networking variables that may be defined as IP addresses or resolvable host names.	No	set Target_IPAddr "192.168.0.10" # *_Broadcast may be an IP address # or resolvable host name if undefined and # if Target_IPAddr is defined as an IP address set Target_BroadcastIPAddr "192.168.0.255" set Target_IEEEAddr "00:90:27:5C:9A:D5" **Default**: A broadcast IP address using 255 in the last segment.
Target_SDR_uC and Target_SDR_uC_Info	Provides SDR-like micro controller information if Target_SDR_Source is set to "Target_Config".	No	set Target_SDR_uC [list BMC CBC] set Target_SDR_uC_Info (BMC,DeviceSlaveAddress) 0x20 set Target_SDR_uC_Info (CBD,DeviceSlaveAddress) 0x28 set Target_SDR_uC_Info(BMC,StringName) "Basbrd Mgmt Ctlr" set Target_SDR_uc_Info(CBC,StringName) "CBC Controller" **Default**: A table based on Platform_Possible_uC.
Target_BMC_SA	specify target BMC slave address	No	Set target_BMC_SA <bmc slave address> **Default:** 0x20

4.5.3.2 Target Configuration Procedures

In addition to the variables listed above, the example target configuration file contains an example of a procedure definition for providing help information encoded in the configuration variables. Once the procedure is defined, it is available for use by the framework and the framework batch files. The following code listing is a procedure definition for `target_help`. This procedure allows ICTS to print the target name and description to the screen:

```
proc target_help { } {
    global Target_Name Target_Description
    ftf_msg "Name:        $Target_Name"
    ftf_msg "Description: $Target_Description"
}
```

4.5.4 User Configuration

The optional user configuration file provides user preference information to the framework. Included with ICTS is a default user configuration file name `i_user.tcl`. This should be suitable for the generic IPMI 1.0/1.5/2.0 system. As with the other configuration files, you may use the template to create a new user configuration file. However, the ICTS graphical user interface provides a faster process for creating a new user configuration file.

The example file is:

```
C:\FTF\templates\config\usercfg.tcl
```

4.5.4.1 Creating a User Configuration File

To create a user configuration file, complete the following procedure:

1. Start up the framework. For information on starting the framework, refer to the
2.

ICTS User Interface section in chapter 3 of this manual or the *Tutorial* chapter of the *IPMI ICTS Developer's Guide*.

3. Load the desired host, platform, and target configuration files. For information on loading configuration files, refer to the *Preparing to Run Tests* chapter of this manual.
4. Load the desired set of test modules. For information on loading test modules, refer to the *Running Conformance Test Suite Tests* chapter of this manual.
5. Using the **Options** and **Display** menus, set preferences for screen font, debug level, verbose level, and so forth.
6. Select the Save User Config... option from the File drop-down menu.

You may edit a new user configuration file to enable options that may not be set from the user interface.

Once created, a user configuration file allows you to load the desired interface and test characteristics you have stored. A saved user file automatically loads the platform file, target file, and your suite of tests.

Place your user configuration files in directory `C:\FTF\users` or any other directory listed by the `Host_UserDirs` variable in your host configuration file.

4.5.4.2 User Configuration Variables

Table 4-18 lists the configuration variables from the `usercfg.tcl` file. Many of these variables are not used by the test modules included in ICTS; however, you may use them while implementing your own test modules. Each variable is accompanied by a description of its use, whether it is required, its default value, and an example:

Table 4-18. Target Configuration Variables

Variable Name	Purpose	Req.	Usage
User_Name	Identifies the owner of this file.	No	set User_Name "Joe User" **Default**: The name of the user configuration file
User_Email User_Pager_Email	User's e-mail address and pager address.	No	set *User_Email joe.user@intel.com* set User_Pager_Email joes-pager@some-pager-service.net **Default**: None
User_Window_Title	FTF window title	No	set User_Window_Title "Joe's window title" **Default**: "FTF - $Target_Name"
User_Revision	Revision number for this file.	No	set User_Revision “0.94” **Default**: 0.0
User_Font	Sets the font in the display window. Variable is a two-item list. The first item is the font family and the second item is the font size.	No	set User_Font [list Fixedsys 10] **Default**: None
User_Debug User_Verbose	Sets the message levels (0-3).	No	set User_Debug 0set User_Verbose 0 **Default**: zero
User_Batch	Enables batch mode.	No	set User_Batch 0 **Default**: zero
User_Email_Okay User *Paging*Okay	Enables automated sending of email and pages.	No	set User_Email_Okay 0 set User_Paging_Okay 0 **Default**: zero
User_Screen_Capture	Enables automatic capture of screen message to the host log file.	No	set User_Screen_Capture 1 **Default**:1
User_Numeric_Prefix	Sets the value of a numeric prefix for use in appending	No	set User_Numeric_Prefix(2) “2#”; # Default is an empty string set User_Numeric_Prefix(16) “0x”; # Default is “0x” **Default**: None

continued

Table 4-18. Target Configuration Variables (continued)

Variable Name	Purpose	Req.	Usage
User_Transports	Specifies transport module priority order.	No	set User_Transports [list ...] **Default**: The order specified in the host configuration file
User_Interface	Specifies default target interface (SMS, I2C, and so forth).	No	set User_Interface SMS **Default**: The first interface listed in the target configuration file
User_Platform	Platform configuration file to load automatically.	No	set User_Platform “Model1000” **Default**: None
User_Target	Target configuration file to load automatically.	No	set User_Target “lab-SC450NX” **Default**: None
User_Tests	List of test modules to load automatically.	No	set User_Tests [list ictsmain] **Default**: None
User_AutoTest	Test to start automatically. Must be a test listed in `User_Tests`.	No	set User_AutoTest ictsmain **Default**: None
User_AutoSetup	Setup file to give to the test specified by `User_AutoTest`.	No	set User_AutoSetup hsetup1 **Default**: None

5 Preparing to Run Tests

This chapter contains information on starting the test framework, exiting the framework, and customizing the framework. Before attempting to run the framework software for the first time, you should be familiar with the information provided in the *Installation and Configuration*, and

ICTS User Interface chapters of this manual.

5.1 Starting the Framework

Before starting the test framework, complete the installation and configuration instructions in the *Installation and Configuration* chapter of this manual. In normal use, the framework can be launched from a batch file or by command line. Once the host system, platform, and target configuration files are customized and placed in their respective directories, use the command line described below to invoke the framework.

5.1.1 Command Syntax

The invocation command line can be placed in a batch file, typed to the Run dialog box or typed to a prompt command line. The `ICTS.BAT` file is included with the framework and, it should be suitable for most cases. If you want to create custom batch files the information in this section may be useful.

The generic name for the Tcl/Tk interpreter command is `wish`. In the case of Tcl 8.3.0 through 8.3.9 (recommended for use with ICTS) the interpreter name is `wish83`. The interpreter allows any `.tcl` file as an argument. However, only the `ftf.tcl` file launches the framework.

Variable setting parameters to `ftf.tcl` override configuration file settings for those variables, and the order of the parameters does not affect precedence.

The syntax described below will bring the framework up on your workstation:

```
Wish83 C:\FTF\base\ftf.tcl [varname[=[varvalue]]] ...
```

Where:

Wish83	The Tcl/Tk interpreter version 8.0.4 or greater. 8.3.0 through 8.3.9.
varname	A Tcl global variable
varvalue	A Tcl global variable

Each *varname/varvalue* combination may take one of three forms:

varname	Sets the named global variable to a value of one.
varname=	Deletes the named global variable.
varname=varvalue	Sets the named global variable to the specified value.

The equal sign syntax causes difficulty in some circumstances, namely in windows shortcuts and in parameters to DOS batch files. To avoid this problem, substitute a double-colon ("::") for the equal sign.

5.1.2 Command Parameters

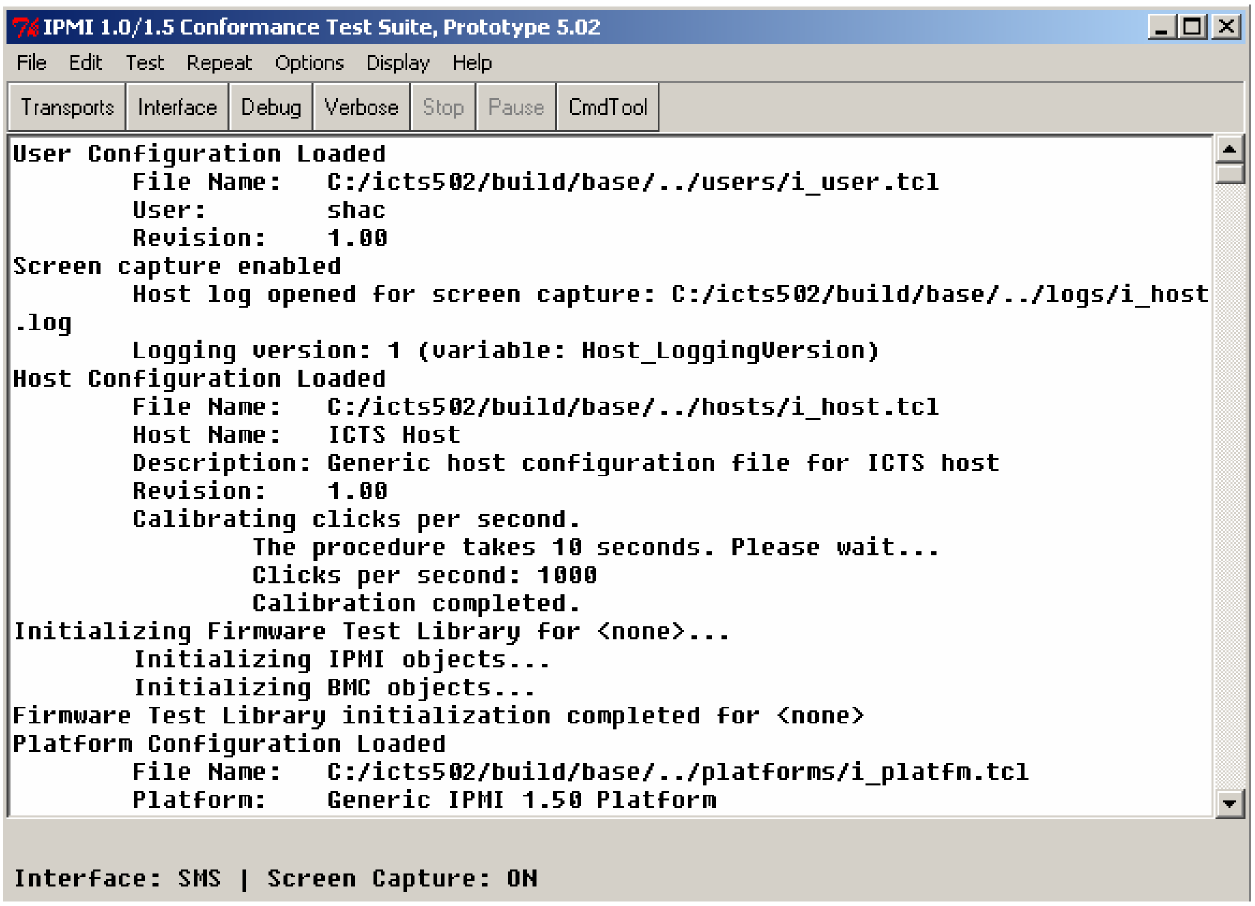
The Tcl global variables described in this section have special meaning only when defined as parameters to `ftf.tcl`. The variables described here may appear only once on the command line. The following table lists the special Tcl global variables by name, type, and purpose:

Table 5-1. Command Parameters

Name	Type	Description
host	File Name	A host configuration file to load after starting the framework. Relative paths are resolved using %ICTSROOOT%\hosts. The default file name extension is .tcl.
platform	File Name	A platform configuration file to load after starting the framework. Requires host. Relative paths are resolved using the Host_PlatDirs variable from the host configuration file. The default file name extension is .tcl.
target	File Name	A target configuration file to load after starting the framework. Requires platform. Relative paths are resolved using the Host_TargDirs variable from the host configuration file. The default file name extension is .tcl.
test	File Name	A test module to load after starting the framework. Requires platform and perhaps target. Relative paths are resolved using the Host_TestDirs variable from the host configuration file. The default file name extension is .tcl.
setup	File Name	A setup file to assign to the test module specified with the test variable. Requires test. Relative paths are resolved using the Host_SetupDirs variable from the host configuration file. The default file name extension is .tcl.
user	File Name	A user configuration file to load after starting the framework. Requires host. Relative paths are resolved using the Host_UserDirs variable from the host configuration file. The default file name extension is .tcl.
resource	File Name	A UNIX†/X-Window style resource data base file.
console	Integer	If non-zero, causes the wish console window to be opened.
debug	Integer (0-3)	The global debug level
verbose	Integer (0-3)	The global verbose level
batch	Integer	If non-zero, enables batch mode.
auto	Integer	If non-zero, auto-starts the test specified by the test variable. Requires test.
help	Integer	If not-zero, causes help to be displayed containing an abbreviated form of the information in this table. (Also available as a Help menu option.)

5.1.3 Example Framework Invocation

This section should contain a complete command line, and a graphic of the results. Probably should be the same graphic that appears in the GUI chapter. Figure 5-1 shows the test framework user interface immediately after invocation. In the example shown, load information for the automatically loaded test libraries appears in the message window.


IPMI 1.0/1.5 Conformance Test Suite, Prototype 5.02
File Edit Test Repeat Options Display Help
Transports Interface Debug Verbose Stop Pause CmdTool

```
User Configuration Loaded
        File Name:   C:/icts502/build/base/../users/i_user.tcl
        User:        shac
        Revision:    1.00
Screen capture enabled
        Host log opened for screen capture: C:/icts502/build/base/../logs/i_host
.log
        Logging version: 1 (variable: Host_LoggingVersion)
Host Configuration Loaded
        File Name:   C:/icts502/build/base/../hosts/i_host.tcl
        Host Name:   ICTS Host
        Description: Generic host configuration file for ICTS host
        Revision:    1.00
        Calibrating clicks per second.
                The procedure takes 10 seconds. Please wait...
                Clicks per second: 1000
                Calibration completed.
Initializing Firmware Test Library for <none>...
        Initializing IPMI objects...
        Initializing BMC objects...
Firmware Test Library initialization completed for <none>
Platform Configuration Loaded
        File Name:   C:/icts502/build/base/../platforms/i_platfm.tcl
        Platform:    Generic IPMI 1.50 Platform

Interface: SMS | Screen Capture: ON
```

Figure 5-1. IPMI 1.0/1.5/2.0 Conformance Test Suite Main Window

5.2 Exiting the Framework

Once the testing session is complete, you can exit the framework by one of several methods. To exit the framework using the File drop-down menu, perform the following procedure:

1. *Click* the File menu. A drop-down menu appears. Figure 5-2 shows the File menu and the Exit selection.

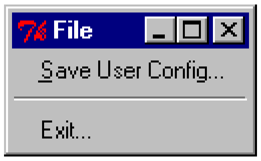


Figure 5-2. File Menu with Exit Selection

2. *Click* Exit. A confirmation dialog box appears. *Click* Yes to confirm or No to continue running the framework. Figure 5-3 shows the Exit Confirmation dialog box.

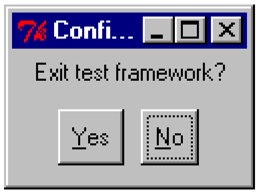


Figure 5-3. Exit Confirmation Dialog Box

In addition to closing the framework by using the Exit command, you can also exit by any of the following methods:

- Double-click the Framework icon in the upper-left corner of the framework window.
- *Click* the X button in the upper-right corner of the window.
- *Click* the Close option from the drop-down menu that appears when you click and hold on the icon in the upper-right corner.

In all cases, the confirmation dialog box shown in Figure 5-3 appears.

5.3 Customizing User Settings

After invoking the user interface, you may want to customize the framework for personal preferences in appearance or for repeated use for testing. The interface allows you to change the appearance of the user interface by sizing the window in the normal click-and-drag Windows NT manner. Additionally, you can change the message text appearance through the Display menu located in the menu bar at the top of the framework window. The Display menu allows you to change the characteristics of your interface. You can change the font choice and size, and you can clear the settings to return to the default. Figure 5-4 shows the Display drop-down menu:

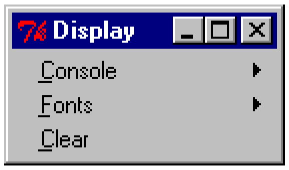


Figure 5-4. Display Drop-down Menu

In addition to changing the appearance of the interface, you can use the Options menu to toggle the screen capture feature on and off, set the global default level of verbose output, change the transport order, and select the default target interface. Figure 5-5 shows the Options drop-down menu:

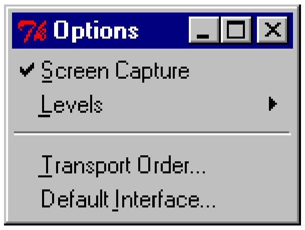


Figure 5-5. Options Drop-down Menu

5.3.1 Changing the Message Window Font

To select a new font, complete the following procedure:

1. *Click* Display in the menu bar.
2. *Click* Fonts. . . The Fonts dialog box appears. Figure 5-6 shows the Fonts dialog box:

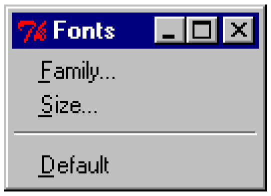


Figure 5-6. Fonts Characteristics Dialog Box

3. *Click* Family. . . The Font Families selection dialog box appears. Figure 5-7 shows the Font Families selection dialog box.

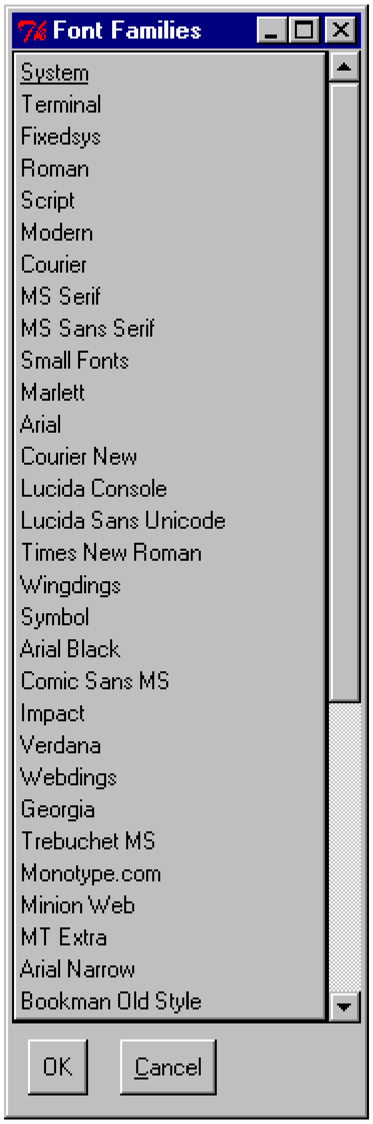


Figure 5-7. Font Family Selection Dialog Box

4. *Click* the font family you desire.
5. *Click* OK. The message window typeface changes to the newly selected font.

5.3.2 Changing the Font Size

To change the size of the font in the message window, complete the following procedure:

1. *Click* Display in the menu bar.
2. *Click* Fonts. . . The Fonts characteristics dialog box appears as shown in Figure 5-6.

3. *Click* Size. . . The Font Sizes selection dialog box appears. Figure 5-8 shows the Font Sizes selection dialog box.

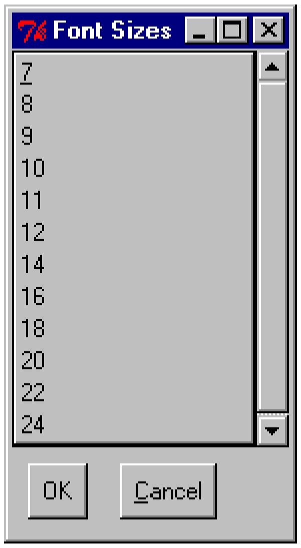


Figure 5-8. Font Sizes Selection Dialog Box

4. *Click* the font size you desire.
5. *Click* OK. The message window text changes to the newly selected size. You may not notice a change if the currently selected font family does not support the selected size. If nothing happens try a different size or a different font family.

5.3.3 Changing Verbose or Debug Output Level

To change the verbose or debug output level, you perform essentially the same procedure. You can change the global output level by beginning the procedure from the Levels item on the Options menu. To change the output level or a particular test and all of its children, begin the procedure from the Levels item on the menu for a selected test.

Figure 5-9 shows the menu for the test, “My Custom Test.” Figure 5-10 shows the Options menu:

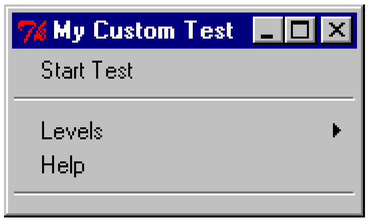


Figure 5-9. Test Menu Showing Levels Item

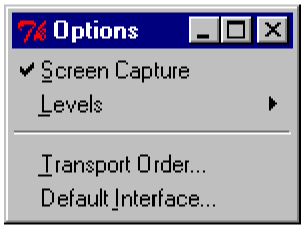


Figure 5-10. Options Menu Showing Levels Item

To change the output level, complete the following procedure:

1. *Click* on the Levels option from the appropriate menu. The Levels drop-down appears. Figure 5-11 shows the Levels drop-down menu.

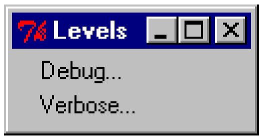


Figure 5-11. The Levels Drop-down

2. *Click* Verbose. . . to change the verbose output level. *Click* Debug. . . to change the debug output level. The Verbose Setting dialog box and the Debug setting dialog box are identical except for their titles and effects. The dialog appropriate to your selection appears. Figure 5-12 shows the Logging dialog box used for setting the verbose and debug levels:

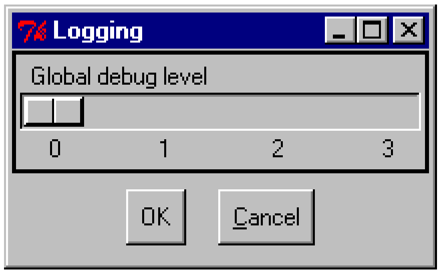


Figure 5-12. Global Debug Dialog Box

3. Select the level of output you desire.
4. *Click* OK to enable the global verbose setting you have selected. *Click* on Cancel to return to the interface without changing the level.

5.3.4 Changing the Order of Transport Modules

If more than one transport module supports a given interface, such as SMS, the priority order feature of the message library for the transport modules determines which module gets responsibility for SMS messages. To change the priority order of the transport modules, complete the following procedure:

1. *Click* Options in the menu bar. The Options drop-down menu shown in Figure 5-5 appears.

2. *Click* Transport Order. . . The Transport Module Order dialog box appears. Figure 5-13 shows the Transport Module Order dialog box with two transport modules listed:

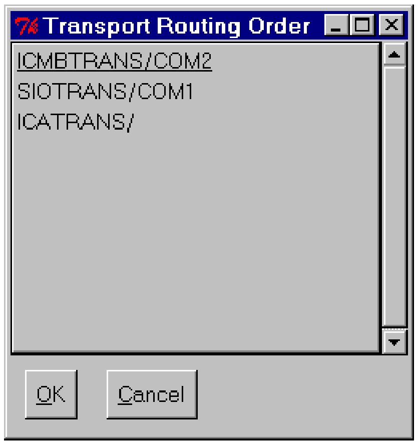


Figure 5-13. Transport Module Order Dialog Box

3. The next time the tests are run, the new high priority module handles messaging for the transport layer.

5.3.5 Changing the Default Target Interface

To change the default target interface, complete the following procedure:

1. *Click* Options in the menu bar. The Options drop-down menu shown in Figure 5-5 appears.
2. *Click* Default Interface. . . The Default Interface dialog box appears. Figure 5-14 shows the Default Interface dialog box with one possible interface listed:

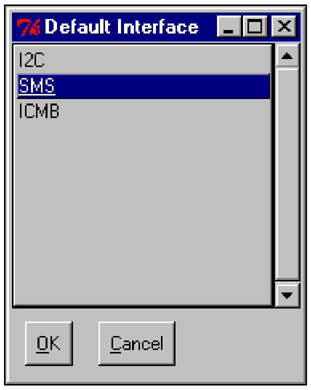


Figure 5-14. Default Interface Dialog Box

3. *Click* the interface item you want to set as the default.
4. *Click* OK to enable the new default interface. *Click* Cancel to return to the interface without change.

5.4 Saving and Retrieving User Settings

Once you have modified the framework to suit your needs and preferences, you can save the settings to the user configuration file. In addition to general appearance settings and the interface and transport settings, saving a user configuration file also saves the list of loaded tests. Invoking the framework with a saved user configuration file automatically loads the tests listed.

To save a user configuration file, complete the following procedure:

1. *Click* File in the menu bar at the top of the framework interface window. The File drop-down menu appears. Figure 5-15 shows the File drop-down menu:

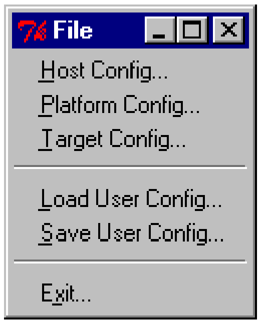


Figure 5-15. File Drop-down Menu

2. *Click* Save User Config. . . saves the user configuration in the file i_user.tcl

5.5 Loading Configuration Files

Between starting the framework and loading test modules, you must load a host configuration file, a platform configuration file, and in most cases a target configuration file. You must load the configuration files in order: host, platform, target. In addition, user-specific information can be loaded through the user configuration file. This section describes how to load the configuration files. For additional information on the contents of the configuration files, refer to the *Installation and Configuration* chapter of this manual. For information about saving a user configuration through the user interface, refer to the *Saving a User Configuration File* section of this chapter.

5.5.1 Host Configuration File

The host configuration file can be loaded from the command line framework invocation or from the graphical interface. A new host configuration file may be loaded at any time and it will override any previously loaded file. This section describes each of these methods for loading the host configuration file.

5.5.1.1 Loading From the Command Line

To load a host configuration file from the command line at a prompt or from a batch file, complete the following procedure:

1. Assign the `host` variable parameter of `ftf.tcl` the name of the host configuration file.
2. Execute the command line, or invoke the batch file.

5.5.1.2 Loading From the User Interface

To load a host configuration file from the graphical user interface, complete the following procedure:

1. From the File drop-down menu, select Load Host Config... The following dialog box appears:

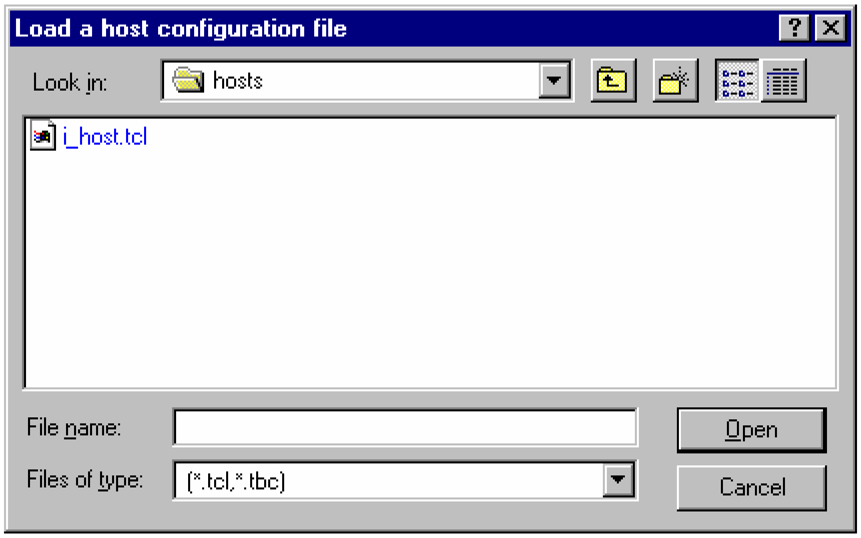


Figure 5-16. Load Host Configuration Dialog Box

2. *Click* on a host configuration file from the browse menu. The starting directory is `C:\FTF\hosts`, but you can use the dialog box's browser to select a file from any disk or directory.
3. *Click* **Open**. The framework will load the selected file and evaluate its contents. The framework will also load any transport modules named in the selected configuration file.

After the successful loading of a host file, the framework allows loading of a platform configuration file or a user configuration file. Until then, the File drop-down menu options platform and user remain unavailable.

5.5.2 Platform Configuration File

Before loading a platform configuration file, you must successfully load a host configuration file. For information on loading a host configuration file, refer to the *Host Configuration File* section of this chapter. Once a host configuration file has been loaded, a new platform configuration file may be loaded at any time. The new file will override any previously loaded file.

A platform configuration file may be loaded by assigning a value to the `platform` variable in the `ftf.tcl` invocation, by invoking the file selection dialog box from the File drop-down menu, and by defining the platform configuration file in a user configuration file. This section describes each of these three methods for loading a platform configuration file.

5.5.2.1 Loading From the Command Line

To load a platform configuration file from the command line at a prompt or from a batch file, complete the following procedure:

1. Assign the `platform` variable parameter of `ftf.tcl` the name of the platform configuration file.
2. Execute the command line, or invoke the batch file.

5.5.2.2 Loading From the User Interface

To load a host configuration file from the graphical user interface, complete the following procedure:

1. From the File drop-down menu, select Load Platform Config... Except in name, the Load Platform Config... dialog box is identical to the Load Host Config... dialog box shown in Figure 5-16.
2. *Click* on a platform configuration file from the browse menu. The starting directory is the first directory listed in the `Host_PlatDirs` variable of the host configuration file`,` but you can browse to select a file from any disk or directory.
3. *Click* Open. The framework loads the selected file and evaluates its contents.

After the successful loading of a platform file, the framework allows loading of a target configuration file. Until then, the File drop-down menu option Target remains unavailable Also, if a platform configuration file contains the required default settings to make a target configuration file unnecessary, the framework enables the Test drop-down menu Load... option. You may then begin loading test modules.

5.5.3 Target Configuration File

Before loading a target configuration file, you must successfully load a platform configuration file. For information, refer to the *Platform Configuration File* section of this chapter. Once a platform configuration file has been loaded, a new target configuration file may be loaded at any time. The new file will override any previously loaded file.

A target configuration file may be loaded by assigning a value to the `target` variable in the `ftf.tcl` invocation, by invoking the file selection dialog box from the File drop-down menu, and by defining the target configuration file in a user configuration file. This section describes each of these three methods for loading a target configuration file.

5.5.3.1 Loading From the Command Line

To load a target configuration file from the command line at a prompt or from a batch file, complete the following procedure:

1. Assign the `target` variable parameter of `ftf.tcl` the name of the target configuration file.
2. Execute the command line, or invoke the batch file.

5.5.3.2 Loading From the User Interface

To load a host configuration file from the graphical user interface, complete the following procedure:

1. From the File drop-down menu, select Load Target Config.... Except in name, the Load Platform Config... dialog box is identical to the Load Host Config... dialog box shown in Figure 5-16.
2. *Click* a target configuration file from the browse menu. The starting directory is the first directory listed in the `Host_TargDirs` variable of the host configuration file, but you can browse to select a file from any disk or directory.
3. *Click* Open. The framework loads the selected file, evaluates its contents, and opens the various ports to provide messaging access to the target machine.

After the successful loading of a target file, the framework allows loading of test modules. Until then, the Test drop-down menu Load... option remains grayed out unless the platform configuration file allows loading of test modules as described in the *Platform Configuration File* section of this chapter.

5.5.4 User Configuration File

User configuration files load user preferences, test modules, and specified host, platform, and target configuration files. For information on defining configuration files and test modules in the user configuration file, refer to the *Installation and Configuration* chapter of this manual. A new user configuration file may be loaded at any time. The new file will override any previously loaded file; however, settings not specified in the new file retain values from the old file.

A user configuration file may be loaded by assigning a value to the `user` variable in the `ftf.tcl` invocation or by invoking the file selection dialog box from the File drop-down menu. This section describes each of these three methods for loading a user configuration file.

5.5.4.1 Loading From the Command Line

To load a user configuration file from the command line at a prompt or from a batch file, complete the following procedure:

1. Assign the `user` variable parameter of `ftf.tcl` the name of the user configuration file.
2. Execute the command line, or invoke the batch file.

5.5.4.2 Loading From the User Interface

To load a host configuration file from the graphical user interface, complete the following procedure:

1. From the File drop-down menu, select the Load User Config... option. Except in name, the Load Platform Config... dialog box is identical to the Load Host Config... dialog box shown in Figure 5-16.
2. *Click* a user configuration file from the browse menu. The starting directory is the first directory listed in the `Host_UserDirs` variable of the host configuration file, but you can browse to select a file from any disk or directory.
3. *Click* Open. The framework loads the selected file, evaluates its contents, and loads defined platform configuration files, target configuration files, and test modules.

After successfully loading a user configuration file, the framework takes on the characteristics defined in the file. All test modules defined in the configuration file become active, and testing can begin.

5.5.5 Saving a User Configuration File

After initially loading all the necessary configuration files, customizing your framework environment, and loading test modules, you can save the state of the system to the user configuration file. Once a snapshot of the system is saved, the user configuration file loads the configuration files, sets the environment options, and loads the test modules automatically.

You can save a new user configuration file any time after successfully loading a host configuration file. To save a user configuration file, complete the following procedure:

1. From the File drop-down menu, select Save User Config... It saves the user configuration in the file i_user.tcl

After creating and saving a user configuration file, you can edit it to customize settings. For additional information about the contents of a user configuration file, refer to the *Installation and Configuration* chapter of this manual.

5.6 Loading Tests

Once the framework is invoked and all configuration files are loaded, you may begin loading test modules. This section describes the process of loading test modules. For information about creating and loading configuration files, refer to the *Installation and Configuration* chapter of this manual. For information on creating test modules, refer to the *Intelligent Platform Interface (IPMI) Conformance Test Suite (ICTS) Developer's Guide*.

5.6.1 Test Modules

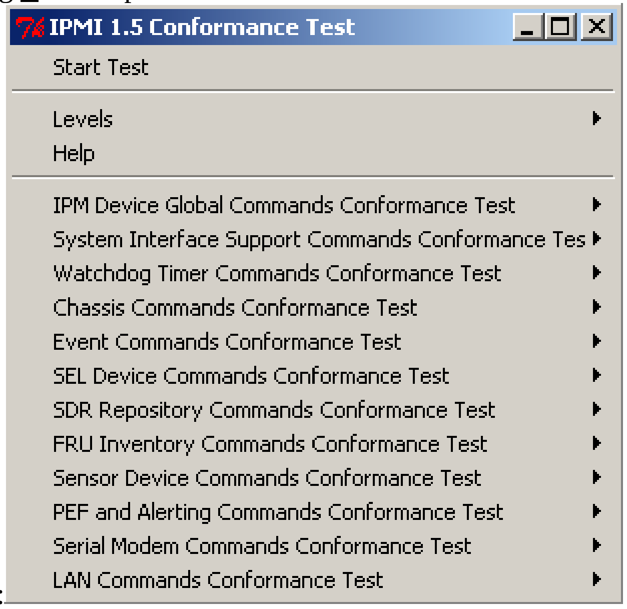
Test modules may be loaded or reloaded at any time. Reloading a test replaces the preciousprevious instance of itself, destroying any information previously associated with the test. Loading new tests has no affect on previously loaded test modules.

NOTE

Excessive reloading of test modules exposes a bug in the GUI menu management code of the Win32 implementation of Tcl/Tk. This most often manifests itself as a General Protection fault when you exit the framework. Other oddities, such as erratic behavior of the cascading menu items, may be observed.

Once the framework loads a test module to memory, the Test drop-down menu shows a new line item for that module. If the module has built-in help, the framework adds an item to the Help menu. Any loaded test may load additional test modules as children. Unless the parent test specifies otherwise, child tests appear in the Test menu hierarchy under the parent's menu entry. Child tests may also have entries in the Help menu.

The following diagram shows the cascading Test drop-down menu after a series of test modules and test module children have been loaded:


IPMI 1.5 Conformance Test
Start Test
Levels
Help
IPM Device Global Commands Conformance Test
System Interface Support Commands Conformance Tes
Watchdog Timer Commands Conformance Test
Chassis Commands Conformance Test
Event Commands Conformance Test
SEL Device Commands Conformance Test
SDR Repository Commands Conformance Test
FRU Inventory Commands Conformance Test
Sensor Device Commands Conformance Test
PEF and Alerting Commands Conformance Test
Serial Modem Commands Conformance Test
LAN Commands Conformance Test

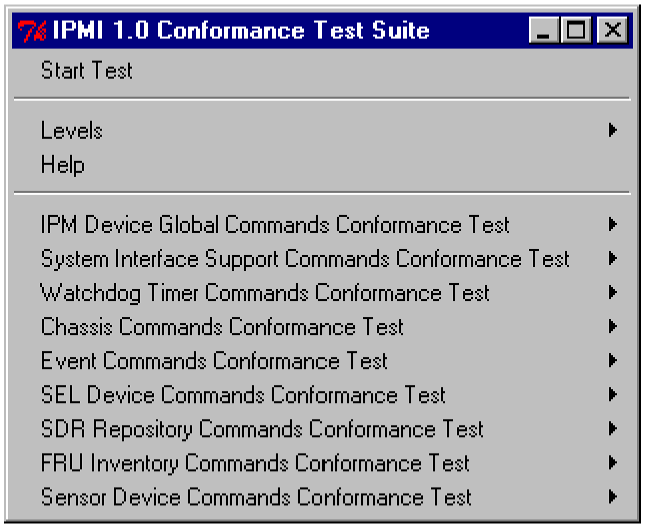

IPMI 1.0 Conformance Test Suite
Start Test
Levels
Help
IPM Device Global Commands Conformance Test
System Interface Support Commands Conformance Test
Watchdog Timer Commands Conformance Test
Chassis Commands Conformance Test
Event Commands Conformance Test
SEL Device Commands Conformance Test
SDR Repository Commands Conformance Test
FRU Inventory Commands Conformance Test
Sensor Device Commands Conformance Test

Figure 5-17. Test Menu With Loaded Tests

There are three methods for loading test modules:

1. A single test module can be loaded in the `wish83 ftf.tcl` command line.
2. Multiple tests can be loaded from the Test drop-down menu.
3. Multiple tests can be loaded from the user configuration file.

5.6.1.1 Command Line Test Loading

The `wish83ftf.tcl` command line syntax allows assigning of a test module name to the `test` variable. Only one test may be loaded from the command line. However, additional test modules may be loaded by spawning children from the test specified in the command line.

To load a single test module from the command line at either the prompt or from a batch file, complete the following procedure:

1. Access the prompt or open the appropriate batch file.
2. Type a command line that conforms to the syntax specifications described in the *Command Syntax* section of this chapter.
3. Assign a test module name to the `test` variable.

5.6.1.2 Loading Tests From The Test Menu

When loading test modules from the framework user interface, a browsing dialog box allows multiple selections. Each test may spawn additional tests.

To load a test from the user interface, complete the following procedure:

1. From the Test drop-down menu, select Load.... A dialog box containing a file list from the directory listed in the `Host_TestDirs` variable of the host configuration file appears. By browsing, you can select test modules from any disk or directory.
2. *Click* the test module or modules you want to load.
3. *Click* Load/OK button. The framework loads the selected test modules, and they appear in the test list. Figure 5-18 shows the Test menu with the newly loaded test, “My Custom Test:”

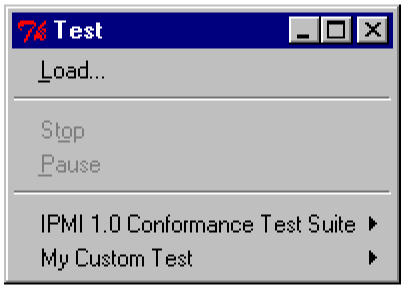


Figure 5-18. The Test Menu with "My Custom Test" Loaded

5.6.2 Loaded Test Menu Items

Each test listed in the Test menu hierarchy has a list of cascading options. This section lists and describes the possible menu items for loaded tests. After traversing the menus for a test anand all of its children to the lowest level, no additional tests appear in the Test menu. At each level, the same options appear in the Test menu for the currently selected test. Figure 5-19 shows the options for My Custom Test. Because My Custom Test spawns no child test, no additional tests appear in the menu:

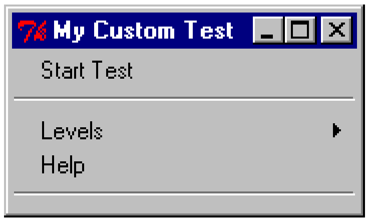


Figure 5-19. Available Options for a Loaded Test

The following is a list of the possible menu items associated with a loaded test:

- Start Test – If the test module contains a `test_main` procedure, this item appears. If the test requires a setup file, this item appears but is unavailable. If the test contains a `test_setup` procedure, this item remains dimmed until `test_setup` succeeds.
- Levels – This item always appears in the menu. It expands to provide menu options for setting local debug and verbose levels.
- Help – If the test module contains a `test_help` procedure, this item appears in the menu.
- Child tests – If the test has children, each appears as a separate cascading menu.

5.6.3 Setup Files

You may pass a setup file to a test module any time after a test module has been loaded successfully. Some test modules require setup files, in which case the Start Test entries on their cascading menus will be unavailable. There are two ways to specify a setup file for a test module:

- With the `setup` variable as a parameter to `ftf.tcl`, perhaps done by a batch file. Only one setup file may be loaded with this method and it applies only to the test module loaded with the `test` variable.
- From the Setup... option of the test's cascade menu.
- Via a user configuration file.

In the case of the menu option, you will be presented with a file dialog box for selecting a file. The starting directory is the first listed in the `Host_SetupDirs` variable of the host configuration file, but you can use the navigation features to select from any disk or directory.

After you select a file, it is loaded into the same space as the test with which it is to be associated. Thus, any variables and procedures defined in the setup file become visible to the test module. Any variables and procedures already defined by the test module are replaced if there are variables or procedures of the same name in the setup file.

After the successful loading of a setup file, the Start Test and/or Start Tool options on the test's cascade menu become enabled if they were previously unavailable.

You may load a new setup file at any time. Any variables or procedures defined in the new setup file replace those defined in the old setup file, but any settings left unspecified in the new setup file are retained from the old file.

6 Running Conformance Test Suite Tests

This chapter describes methods of running loaded test modules and analyzing the results. Before running a test module, the system must be installed and configured. For additional information on installing the framework and creating the necessary configuration files, refer to the *Installation and Configuration* chapter of this manual. For information on creating test modules, refer the *Intelligent Platform Management Interface (IPMI) Conformance Test Suite Developer's Guide*.

In most cases a test (or tool) may be started immediately after loading. For information on loading test modules, refer to the *Preparing to Run a Test* chapter of this manual.

6.1 Running the Entire Set of Tests

While it is only possible to execute one test through the framework, that test may run other tests. A test that executes other tests is called a parent. The tests a parent runs are called child tests. The children may have children of their own. This hierarchy of parents and children is duplicated in the GUI menus described in the

ICTS User Interface chapter of this manual. Running the test at the top of the hierarchy results in execution of all child tests, and thus the entire test suite.

Navigating the test hierarchy downward through the menus allows you to run any subset of the entire suite. At the bottom of the hierarchy you can run individual tests. Child tests are defined in the parent tests, and in the case of short versions of ICTS, children are hard-coded into the framework hierarchy. It is possible in FTF to write a parent test that determines its child tests on-the-fly, perhaps based on input from the user or a configuration file, but the non-developer's version of ICTS does not allow this.

To run a group of framework tests, complete the following procedure:

1. *Click* Test in the menu bar at the top of the framework user interface window. The Test drop-down menu appears. Figure 6-1 shows the Test drop-down menu:

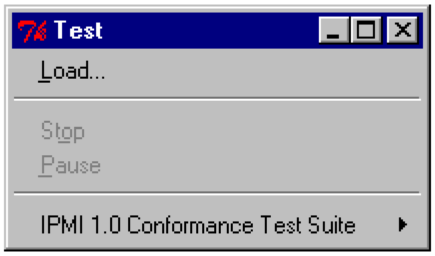
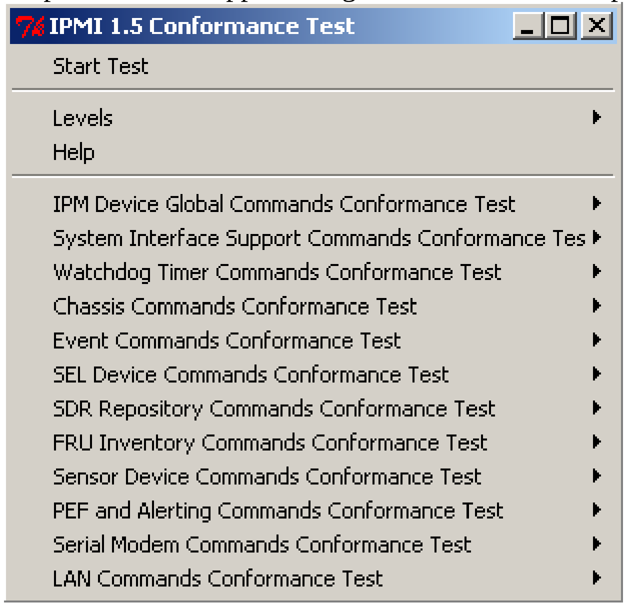
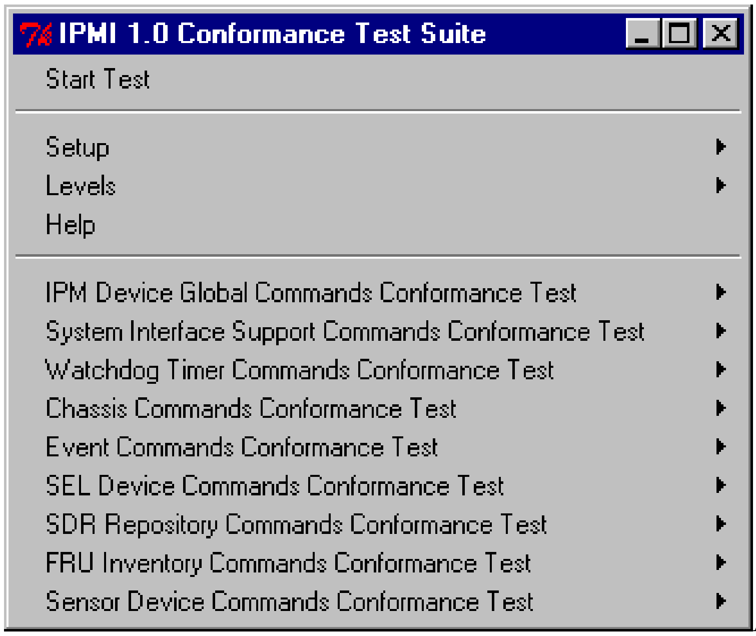


Figure 6-1. Test Drop-down Menu

2. Select IPMI 1.0/1.5/2.0 Conformance Test Suite. The IPMI Conformance Test Suite drop-down menu appears. Figure 6-2 shows the drop-down menu:


IPMI 1.5 Conformance Test
Start Test
Levels
Help
IPM Device Global Commands Conformance Test
System Interface Support Commands Conformance Tes
Watchdog Timer Commands Conformance Test
Chassis Commands Conformance Test
Event Commands Conformance Test
SEL Device Commands Conformance Test
SDR Repository Commands Conformance Test
FRU Inventory Commands Conformance Test
Sensor Device Commands Conformance Test
PEF and Alerting Commands Conformance Test
Serial Modem Commands Conformance Test
LAN Commands Conformance Test


IPMI 1.0 Conformance Test Suite
Start Test
Setup
Levels
Help
IPM Device Global Commands Conformance Test
System Interface Support Commands Conformance Test
Watchdog Timer Commands Conformance Test
Chassis Commands Conformance Test
Event Commands Conformance Test
SEL Device Commands Conformance Test
SDR Repository Commands Conformance Test
FRU Inventory Commands Conformance Test
Sensor Device Commands Conformance Test

Add the diaglog box for IPMI 1.5 platfom also.

Figure 6-2. IPMI 1.0/1.5/2.0- Conformance Test Suite Drop-down Menu

3. Select the test group you want to load. A list of the tests in the group appears. Figure 6-3 shows the list of tests for the IPM Device Global Command Conformance Test:

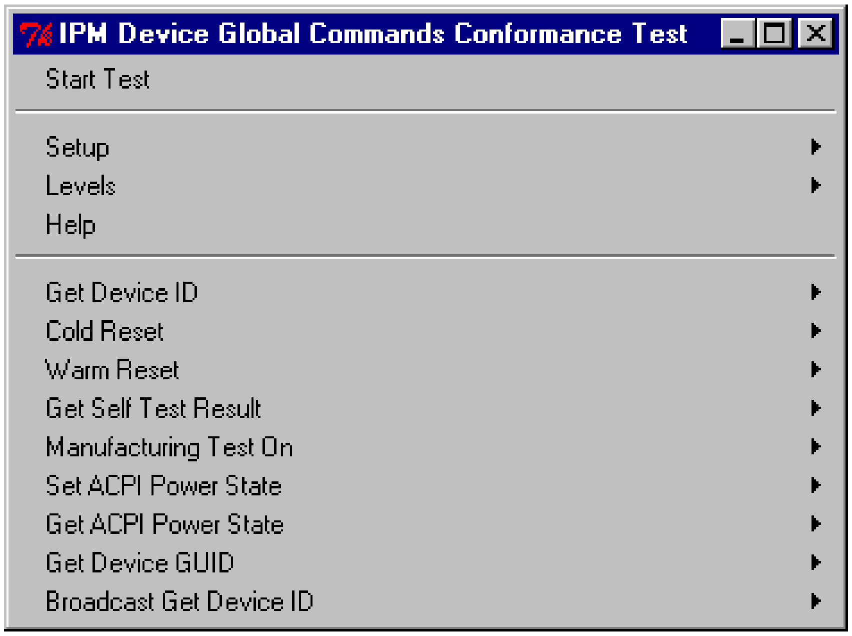


Figure 6-3. IPM Device Global Command Conformance Test Drop-down Menu

4. Select an individual test from the list. The Test menu for the selected test appears. Figure 6-4 shows the Test menu for the Get Device ID test:

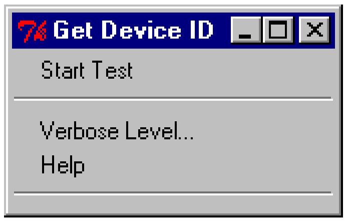


Figure 6-4. Get Device ID Dialog Box

5. Select Start Test.. . . The framework begins the test and responses begin to appear in the framework message window.

6.2 Stopping or Pausing Tests

In many cases, it is possible to stop or pause a test in progress. However, not all tests support stops and pauses. Tests supporting safe stop and pause points make periodic API calls to the framework to seek the stop or pause request. Tests that do not make API calls cannot receive start and stop requests. This disables the framework's user interface for the duration of the test by preventing the Tcl/Tk interpreter from processing events in its event queue.

To stop a stop-enabled test, complete the following procedure:

1. From the Test drop-down menu, select Stop. There is no confirmation dialog box. A stop request is returned to the test, but the test must have the characteristics necessary to stop itself.

To pause a pause-enabled test, complete the following procedure:

1. From the Test drop-down menu, select Pause. A resume dialog box appears.
2. Resume the test or stop the test by clicking the approprite button.

7 ICTS User Interface

The Intelligent Platform Management (IPMI) Conformance Test Suite (ICTS) consists of a host-executable graphical user interface to allow configuration of conformance tests, running of tests, and viewing of test results. The interface consists of a message window, a status bar, and menus. This chapter provides an overview description of the interface window and menus. For information on launching the test framework and on using the dialog boxes for configuration, testing and analysis, refer to the *Installation and Configuration* chapter of this manual and to the *Tutorial* chapter of the *Intelligent Platform Management (IPMI) Conformance Test Suite (ICTS) Developer's Guide*.

7.1 Windows

Each window or menu has a specific purpose. A description of each appears in this chapter, and every section contains a description of the window or menu and provides a graphic of its appearance during typical use.

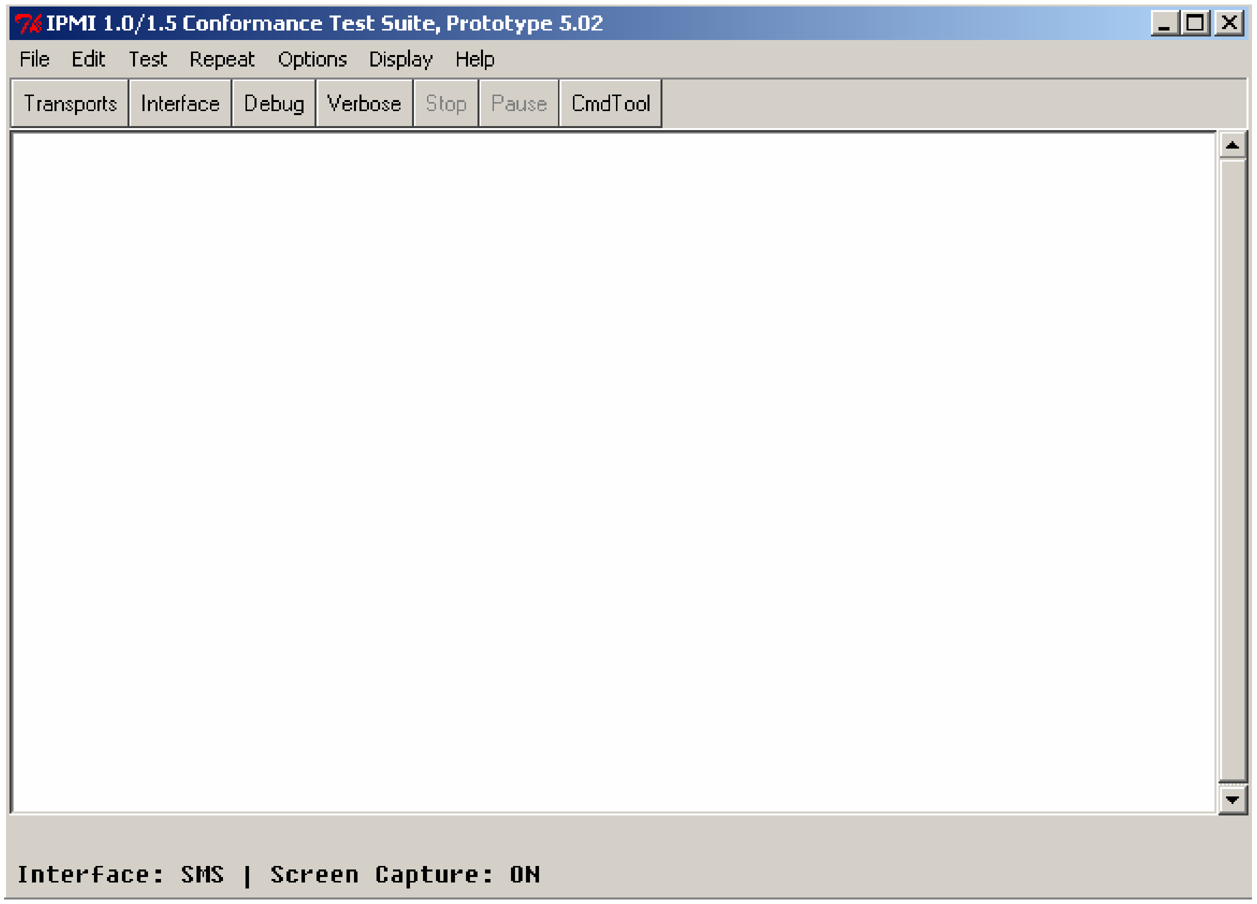
Table 7-1 contains a list of the windows and their purposes.

Table 7-1. ICTS Graphical User Interface Windows

Screen Object Name	Purpose
Main Message Window	Provides space for viewing messages generated by configuration, loading, testing, and analysis. The menu bar at the top provides access to the other menus.
Menu Bar	Provides access to drop-down menus.
Status Bar	Displays messages from test or from the framework.

7.1.1 Main Window

The main ICTS window provides access to menus, system information, and test feedback. Figure 5-1 depicts the main window immediately after launch. Diagnostic messages for configuration file-loaded tests have scrolled past, and the status bar at the bottom of the window shows the framework's initial state.



Interface: SMS | Screen Capture: ON

Figure 7-1. IPMI Conformance Test Suite Main Window

7.1.2 Status Bar

The Status Bar at the bottom of the main window shows general information about the user interface, including the target configuration, and the status of the screen capture utility. Additional information appears according to changes in the status of the framework.

7.1.3 Menus

Drop-down menus conform to Microsoft Windows NT graphical user interface specifications. Each menu has a title in the menu bar at the top of the main window area. Menu commands are organized by type under menu titles. Click and hold on the menu title to access the menu items. Positioning the cursor over the menu command in the drop-down list and releasing the mouse button invokes the command item. Menu items followed by ellipses (. . .) invoke dialog boxes.

Every menu and menu command has a corresponding keystroke. Invoke drop-down menus by pressing the *ALT* key plus the underlined letter of the menu title. For example, to display the **File** drop-down menu, hold *ALT* and press *F*. Once a menu is invoked, pressing the underlined letter of any command listed in the drop-down executes that command. The *ESC* key cancels drop-down access.

This chapter contains a section describing each drop-down menu and its commands. Each section shows the screen during normal use. Table 7-2 shows the menu titles, their purposes, and the keystrokes for access.

Table 7-2. ICTS Graphical User Interface Menus

Menu Name	Keystroke	Purpose
File	ALT+F	Loading and saving configuration files.
Edit	ALT+E	Copying selected text.
Test	ALT+T	Loading, pausing, and stopping a test module.
Repeat	ALT+R	Repeating the last test run or the last tool used.
Options	ALT+O	Configuring debug levels, message routing, interface type, data source, and microcontroller type.
Display	ALT+D	Configuring window content and appearance.
Help	ALT+H or F1	Providing support information.

7.1.3.1 File Menu

The File drop-down menu gives access to dialog boxes that let you load configuration files for the host environment, the platform, the target, and the user. Additionally, the **File** menu offers access to a dialog box for modifying and saving user configuration files. Figure 7-2 shows the contents of the **File** drop-down menu:

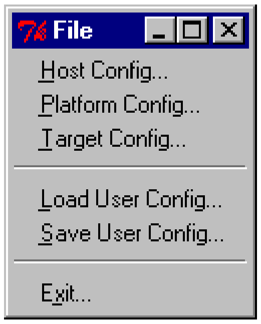


Figure 7-2. File Menu

The following list contains the menu items offered in the **File** menu along with a function description of each:

- Host Config...—Presents a dialog box for loading a host configuration file. Figure 7-3 shows the Host Config... dialog box:

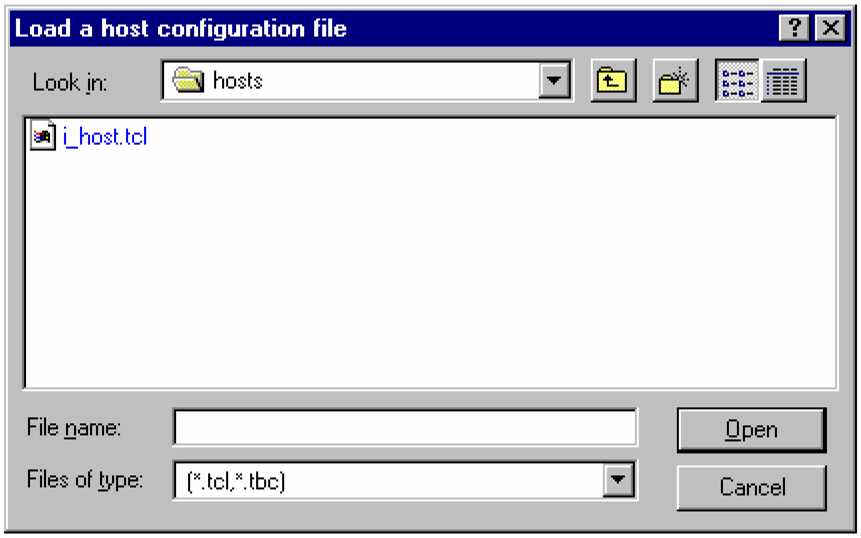


Figure 7-3. Load a Host Configuration File Dialog

- Platform Config...—Presents a dialog box for loading a platform configuration file. This menu item remains unavailable until a host configuration file is successfully loaded. Except in name, the Platform Config... dialog box is identical to the box shown in Figure 7-3.
- Target Config...—Presents a dialog box for loading a target configuration file. This menu item remains unavailable until a platform configuration file is successfully loaded. Except in name, the Target Config... dialog box is identical to the box shown in Figure 7-3.
- Load User Config...—Presents a dialog box for loading a user configuration file. This menu item remains unavailable until a host configuration file is successfully loaded. Except in name, the Load Config... dialog box is identical to the box shown in Figure 7-3.

Save User Config... —saves the user configuration to the file i_user.tcl

- Exit—Exits from the program after presenting a confirmation dialog. Figure 7-4 shows the Exit... dialog box:

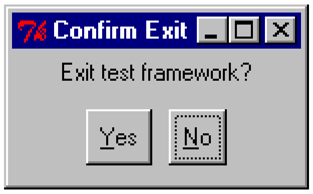


Figure 7-4. Exit Framework Confirmation Box

7.1.3.2 Edit Menu

The Edit drop-down menu contains only the Copy command item. The Copy item remains unavailable, as shown in Figure 7-6, until a block of text is selected. Once text is selected, the Copy item appears as bold and enabled, as shown in Figure 7-5. Selecting the Copy item places the currently selected text in the system clipboard.

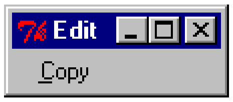


Figure 7-5. Edit Menu Copy-enabled

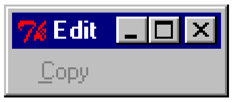


Figure 7-6. Edit Menu Copy-disabled

7.1.3.3 Test Menu

The Test drop-down menu has a unique characteristic. After a test is loaded, an additional menu item representing the newly loaded test appears in the IPMI 1.0/1.5/2.0 Conformance Test Suite drop-down menu. For additional information on using the Test drop-down menu to load, start, and stop tests, refer to *Preparing to Run Tests* chapter of this manual.

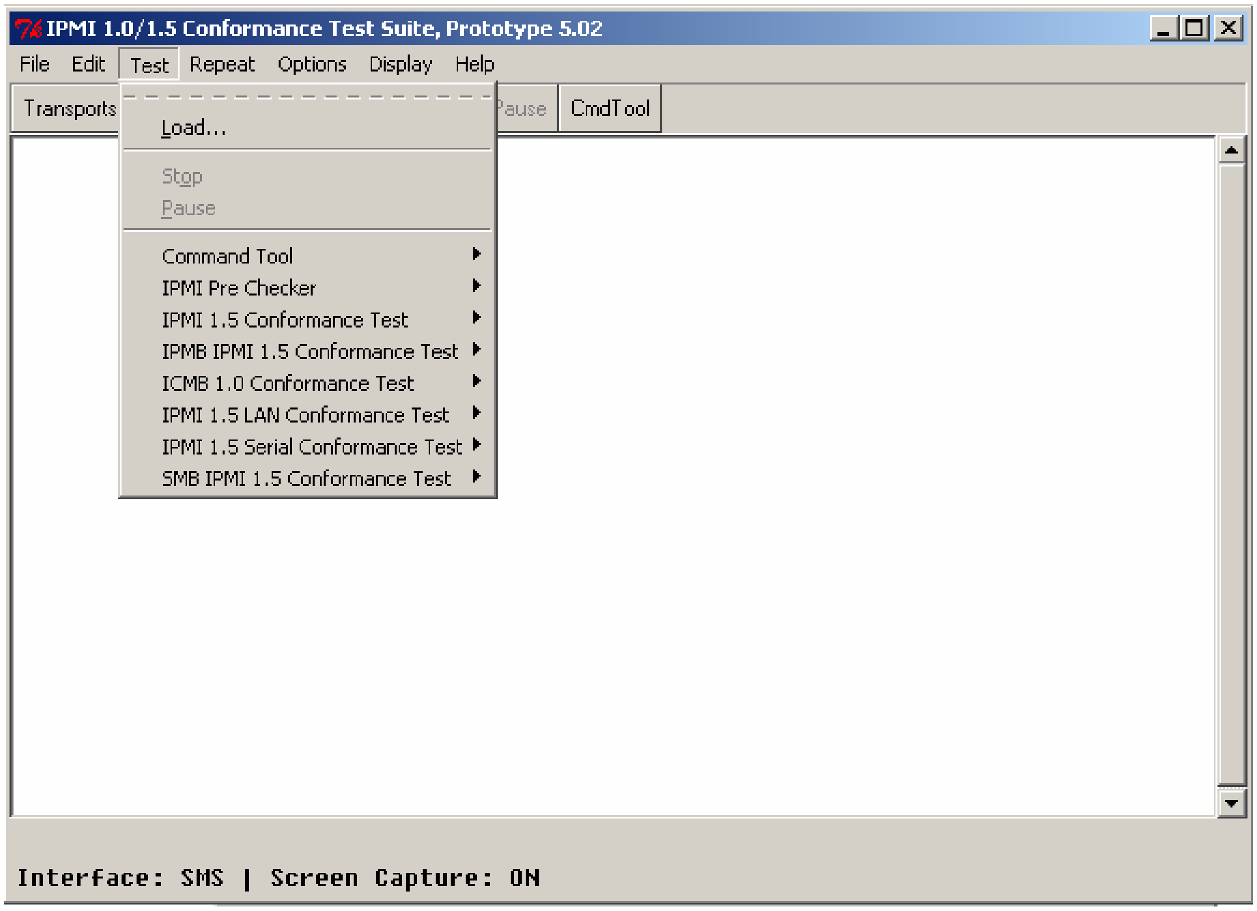
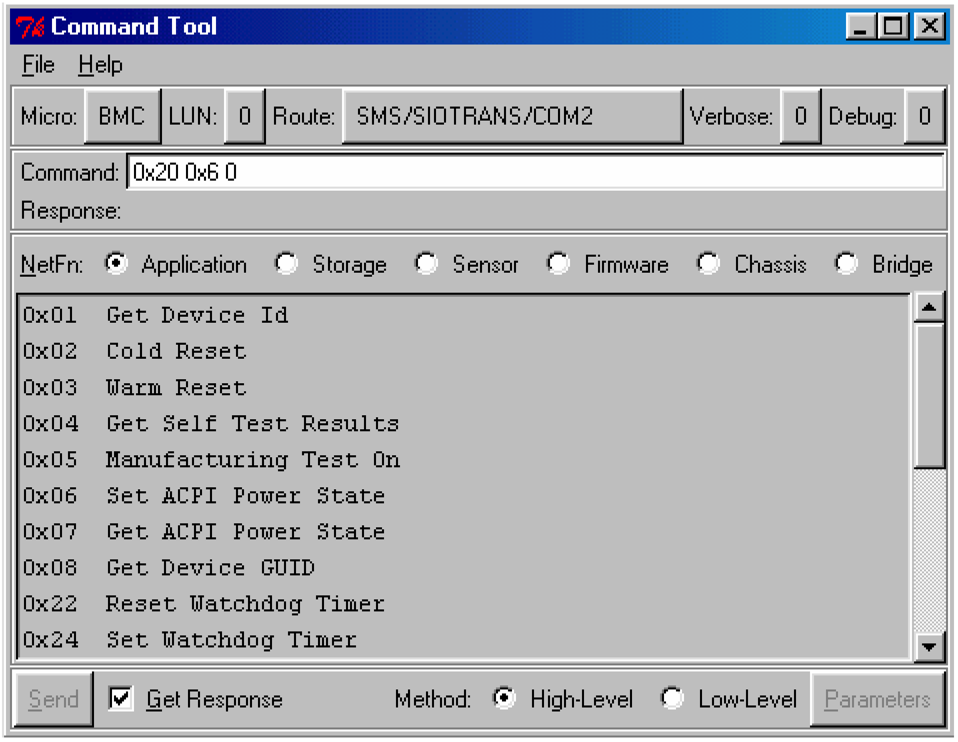

IPMI 1.0/1.5 Conformance Test Suite, Prototype 5.02
File
Edit
Test
Repeat
Options
Display
Help
Transports
Pause
CmdTool
Load...
Stop
Pause
Command Tool
IPMI Pre Checker
IPMI 1.5 Conformance Test
IPMB IPMI 1.5 Conformance Test
ICMB 1.0 Conformance Test
IPMI 1.5 LAN Conformance Test
IPMI 1.5 Serial Conformance Test
SMB IPMI 1.5 Conformance Test
Interface: SMS | Screen Capture: ON

Figure 7-7. Test Menu

The Test drop-down menu contains these items:

- Load. . .—Presents a dialog to allow selection and loading of a test module. This menu item remains dimmed until a target configuration file (or in some cases only a platform configuration file) is successfully loaded. For each loaded test, an item is added to the test menu. Figure 7-7 shows the Test menu with a custom built test named, "*My Custom Test*" loaded:
- Stop—Stops the currently running test or tool, if possible. This menu item is unavailable unless a test or tool is currently running.
- Pause—Pauses the currently running test or tool, if possible. This menu item is dimmed unless a test or tool is currently running.
- Command Tool—



Command Tool

File Help

Micro: BMC LUN: 0 Route: SMS/SIOTRANS/COM2 Verbose: 0 Debug: 0

Command: 0x20 0x6 0

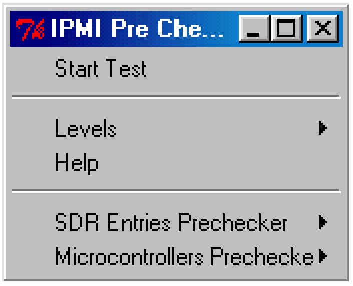
Response:

NetFn: [✓ Application] Storage Sensor Firmware Chassis Bridge

```
0x01   Get Device Id
0x02   Cold Reset
0x03   Warm Reset
0x04   Get Self Test Results
0x05   Manufacturing Test On
0x06   Set ACPI Power State
0x07   Get ACPI Power State
0x08   Get Device GUID
0x22   Reset Watchdog Timer
0x24   Set Watchdog Timer
```

Send [✓ Get Response] Method: [✓ High-Level] Low-Level Parameters

- IPMI Pre Checker—



- IPMI 1.0/1.5/2.0 Conformance Test Suite—Provides access to additional drop-down menus for configuration of tests and access to test data. Figure 7-8 shows the contents of the IPMI 1.0/1.5/2.0 Conformance Test Suite dialog box. The dialog provides access to a series of

drop-down menus through which you can access the automatically loaded test libraries provided by the
framework. For additional information on the drop-down menus in this dialog, refer to the *IPMI Conformance Test Suite Sub-Menus* section of this chapter.

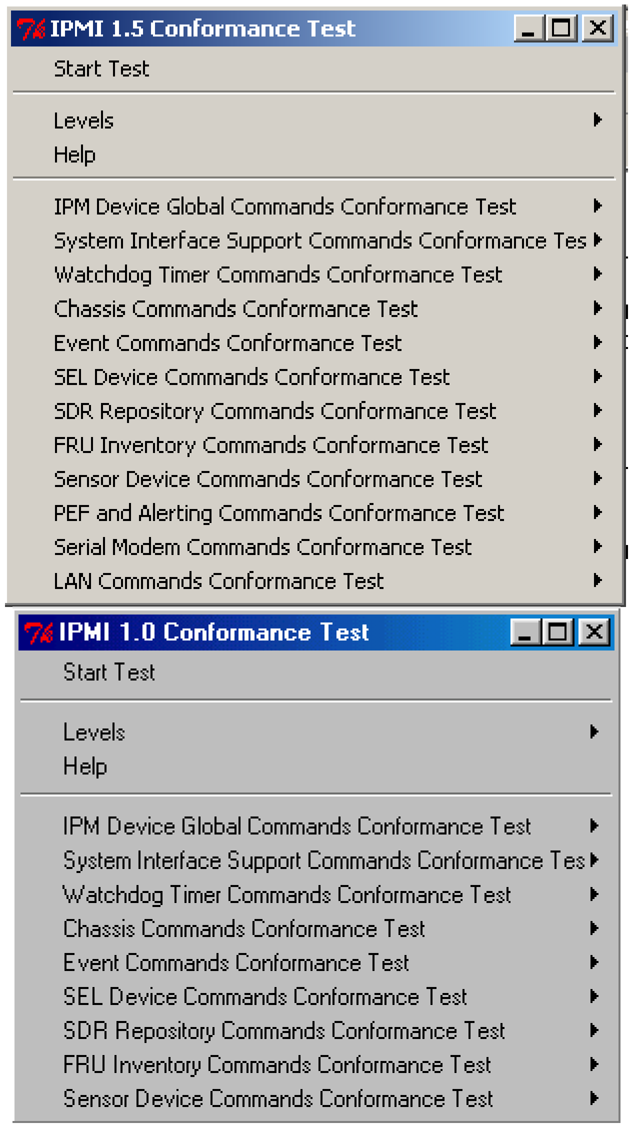
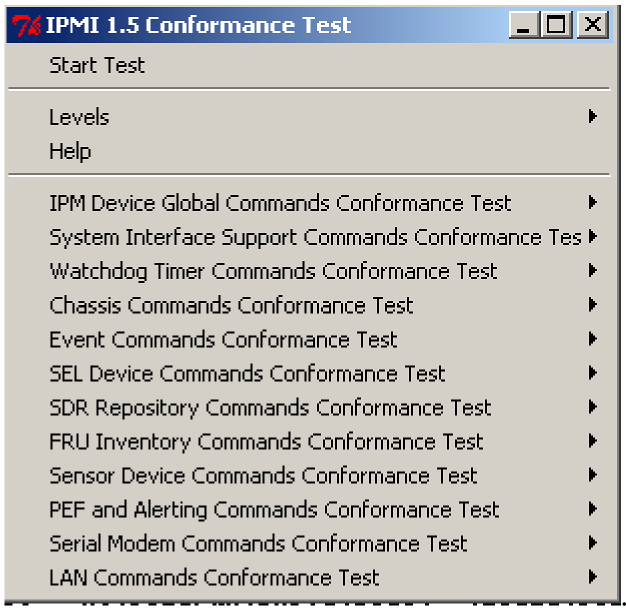
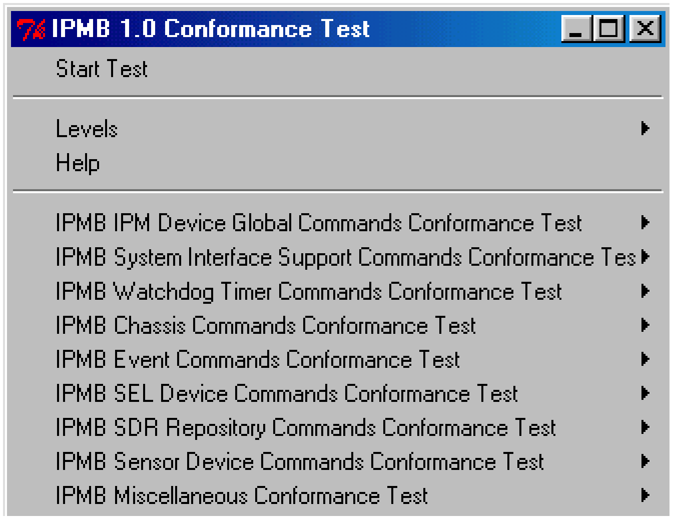

IPMI 1.5 Conformance Test
Start Test
Levels
Help
IPM Device Global Commands Conformance Test
System Interface Support Commands Conformance Tes
Watchdog Timer Commands Conformance Test
Chassis Commands Conformance Test
Event Commands Conformance Test
SEL Device Commands Conformance Test
SDR Repository Commands Conformance Test
FRU Inventory Commands Conformance Test
Sensor Device Commands Conformance Test
PEF and Alerting Commands Conformance Test
Serial Modem Commands Conformance Test
LAN Commands Conformance Test
IPMI 1.0 Conformance Test
Start Test
Levels
Help
IPM Device Global Commands Conformance Test
System Interface Support Commands Conformance Tes
Watchdog Timer Commands Conformance Test
Chassis Commands Conformance Test
Event Commands Conformance Test
SEL Device Commands Conformance Test
SDR Repository Commands Conformance Test
FRU Inventory Commands Conformance Test
Sensor Device Commands Conformance Test

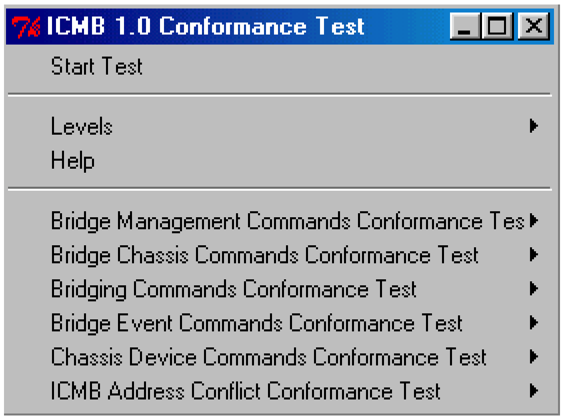
Figure 7-8. IPMI 1.0/1.5/2.0 Conformance Test Suite Drop-down Menus

- IPMB IPMI 1.0/1.5/2.0 Conformance Test—Provides access to additional drop-down menus for configuration of tests and access to test data. Figure 7-8 shows the contents of the IPMI 1.0/1.5/2.0 Conformance Test Suite dialog box. The dialog provides access to a series of drop-down menus through which you can access the automatically loaded test libraries provided by the framework. For additional information on the drop-down menus in this dialog, refer to the *IPMI Conformance Test Suite Sub-Menus* section of this chapter.


IPMI 1.5 Conformance Test
Start Test
Levels
Help
IPM Device Global Commands Conformance Test
System Interface Support Commands Conformance Tes
Watchdog Timer Commands Conformance Test
Chassis Commands Conformance Test
Event Commands Conformance Test
SEL Device Commands Conformance Test
SDR Repository Commands Conformance Test
FRU Inventory Commands Conformance Test
Sensor Device Commands Conformance Test
PEF and Alerting Commands Conformance Test
Serial Modem Commands Conformance Test
LAN Commands Conformance Test


IPMB 1.0 Conformance Test
Start Test
Levels
Help
IPMB IPM Device Global Commands Conformance Test
IPMB System Interface Support Commands Conformance Tes
IPMB Watchdog Timer Commands Conformance Test
IPMB Chassis Commands Conformance Test
IPMB Event Commands Conformance Test
IPMB SEL Device Commands Conformance Test
IPMB SDR Repository Commands Conformance Test
IPMB Sensor Device Commands Conformance Test
IPMB Miscellaneous Conformance Test

- ICMB 1.0 Conformance Test —Provides access to additional drop-down menus for configuration of tests and access to test data. Figure 7-8 shows the contents of the IPMI 1.0/1.5/2.0 Conformance Test Suite dialog box. The dialog provides access to a series of drop-down menus through which you can access the automatically loaded test libraries provided by the framework. For additional information on the drop-down menus in this dialog, refer to the *IPMI Conformance Test Suite Sub-Menus* section of this chapter.


ICMB 1.0 Conformance Test
Start Test
Levels
Help
Bridge Management Commands Conformance Tes
Bridge Chassis Commands Conformance Test
Bridging Commands Conformance Test
Bridge Event Commands Conformance Test
Chassis Device Commands Conformance Test
ICMB Address Conflict Conformance Test

IPMI Conformance Test Suite Sub-Menus

The IPMI 1.0/1.5/2.0 Conformance Test Suite submenus give access to the testing libraries provided by the test framework. Clicking Start Test on the IPMI 1.0/1.5/2.0 Conformance Test Suite menu runs the entire set of loaded tests. Selecting a child test from the test list sub-menu provides access to another test drop-down menu. The new menu appears identical to the one shown in Figure 7-8, except that the list of child tests changes. Each child test menu item provides access to automatically loaded tests. Selecting their respective Start Test item will run that test and its children.

Every test drop-down provides its own Start Test item, as well as Levels and Help items. Levels allows access to debug and verbose level-setting. Help provides information available for the currently loaded tests.

Following the test hierarchy down to a test with no children in the sub-menu allows execution of a single test without initiation of child tests. Figure 7-9 shows the Get Device ID test menu. Get Device ID has no child tests, so there are no additional menu items.

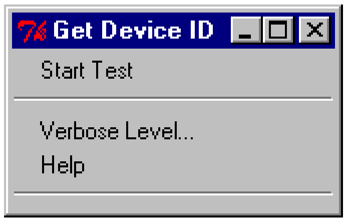


Figure 7-9. Standard Test Dialog

NOTE

The menus for a Custom Test are identical in form and content to the menus and actions for the standard tests described in this section.

7.1.3.4 Repeat Menu

The Repeat drop-down menu allows quick re-invocation of the last test run. Figure 7-10 shows the Repeat menu before a test has been run:

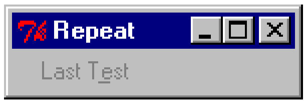


Figure 7-10. Repeat Menu

The Repeat drop-down menu contains these items:

- Last Test—Repeats the most recently completed test. This item remains dimmed until at least one test has successfully completed. During normal operation, the name of the last test run appears in this menu.

After a test has run, the Last Test item is enabled and shows the name of the last test run. Figure 7-11 shows the Repeat menu after the Get Device ID test has run:

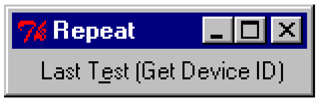


Figure 7-11. Repeat Menu Showing Name of Last Test Run

7.1.3.5 Options Menu

The ICTS Options menu allows customization of the framework user interface behavior. For detailed usage information about the options presented by this menu, refer to the *Installation and Configuration* chapter in this manual.

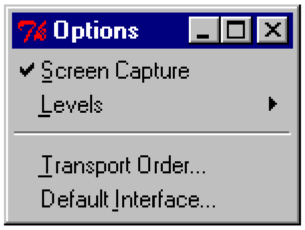


Figure 7-12. Options Menu

The Options drop-down menu contains the following items:

- Screen Capture—Toggles logging of screen messages to the host log file.
- Levels—Displays a cascading menu containing the Debug and Verbose items. Either item presents a dialog box for setting the global level for the selected item. Except for the label, the dialog box for both items appears as shown in Figure 7-14. Figure 7-13 shows the Levels menu:

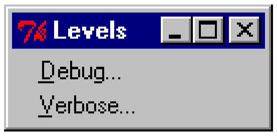


Figure 7-13. Levels Menu Items

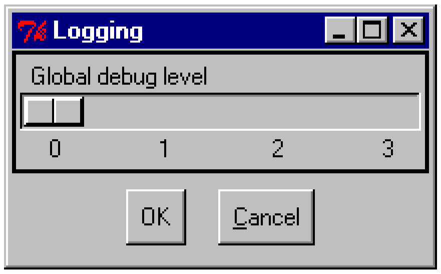


Figure 7-14. The Level Dialog (Global Debug)

- Transport Order. . .—Presents a dialog box for changing the priority order when multiple transport modules are available. Clicking on an item in the dialog promotes that item to the highest priority. Figure 7-15 shows the Transport Module Order dialog box containing two available transport layers:

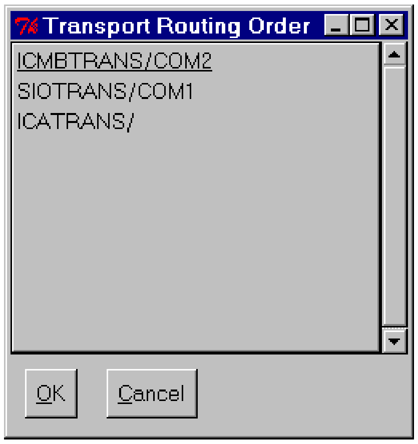


Figure 7-15. Transport Module Order Dialog Box

- Default Interface ...—Presents a dialog box for selecting the default target interface. This dialog lists all open interfaces. Clicking on an item in the list selects that interface. Clicking OK makes the selected interface the default interface. The result appears in the status bar on the main window. Figure 7-16 shows the Default Interface dialog:

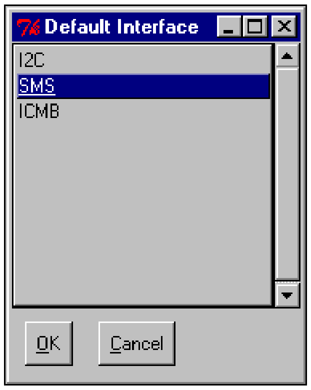


Figure 7-16. Default Interface Dialog Box

7.1.3.6 Display Menu

The Display menu items control the appearance of the ICTS interface window by allowing changes to the appearance of the graphical interface. Figure 7-17 shows the Display drop-down menu:

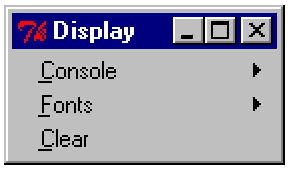


Figure 7-17. Display Menu

The Display drop-down menu contains these items:

- Console—Provides access to the Console menu. The Show option brings up the wish console window (a built-in feature of Tcl/Tk on Win32). The Hide option makes the window disappear. The window allows you to enter Tcl commands. For additional information on use of the Console window, refer to your Tcl/Tk documentation. Figure 7-18 shows the Console menu:

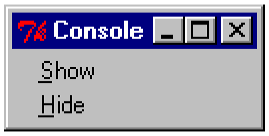


Figure 7-18. Console Menu

Figure 7-19 shows the Console window:

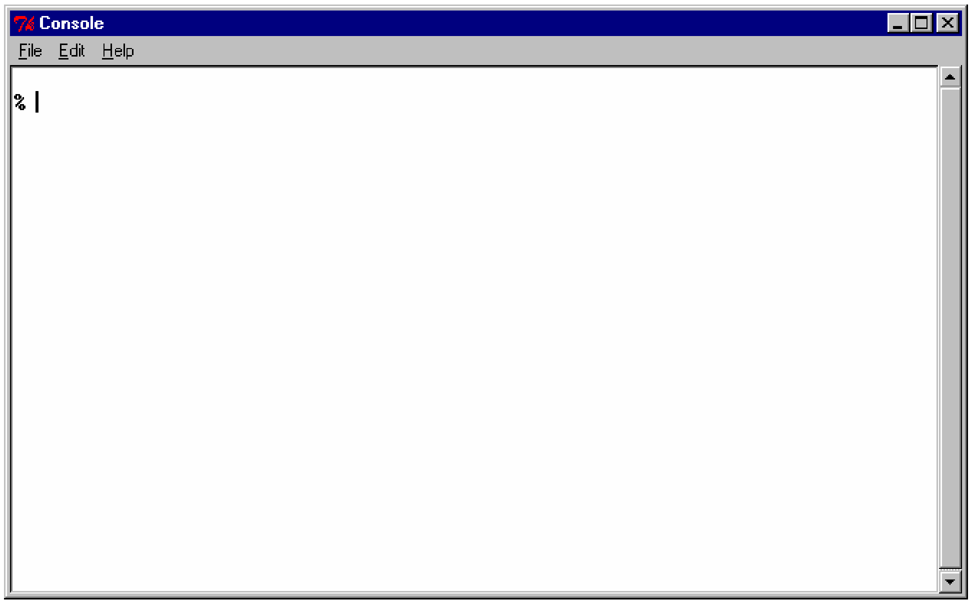


Figure 7-19. Console Window

- Fonts—A cascading menu that provides access to font family selection and font size selection dialog boxes. Selecting the Default item returns the interface to its original state. Figure 7-21 shows the Fonts dialog box:

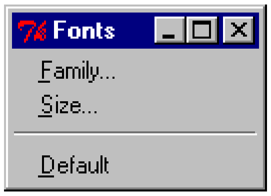


Figure 7-20. Fonts Dialog Box

Selecting the Family. . . item brings the Font Families dialog box up. Selecting a font family from the dialog and clicking the OK changes the font family for the user interface. Figure 7-22 shows the Font Families selection dialog box:

Font Families Selection Window

Selecting the Size. . . item brings the Font Sizes dialog box up. Selecting a size from the dialog and clicking the OK changes the font size for the user interface. Font Sizes selection dialog box Console Window

- Fonts—A cascading menu that provides access to font family selection and font size selection dialog boxes. Selecting the Default item returns the interface to its original state. Figure 7-21 shows the Fonts dialog box:

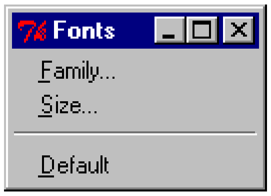


Figure 7-21. Fonts Dialog Box

Selecting the Family. . . item brings the Font Families dialog box up. Selecting a font family from the dialog and clicking the OK changes the font family for the user interface. Figure 7-22 shows the Font Families selection dialog box:

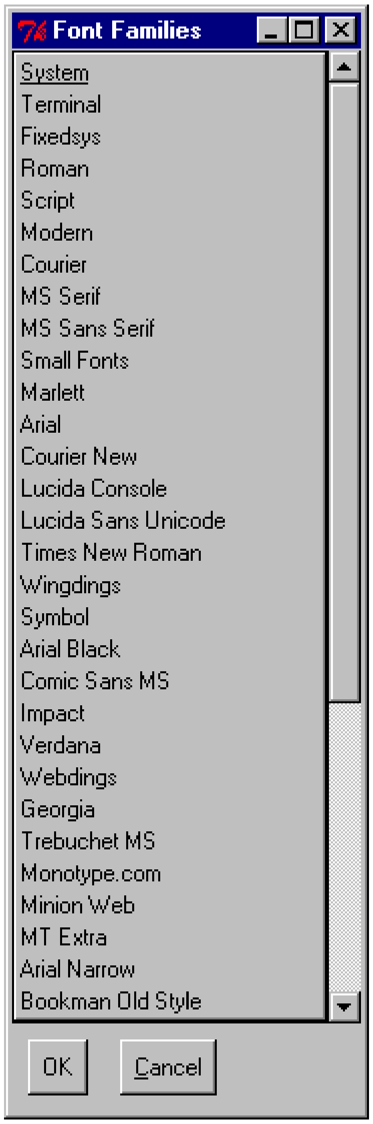


Figure 7-22. Font Families Selection Window

Selecting the Size. . . item brings the Font Sizes dialog box up. Selecting a size from the dialog and clicking the OK changes the font size for the user interface. Figure 7-23 shows the Font Sizes selection dialog box:

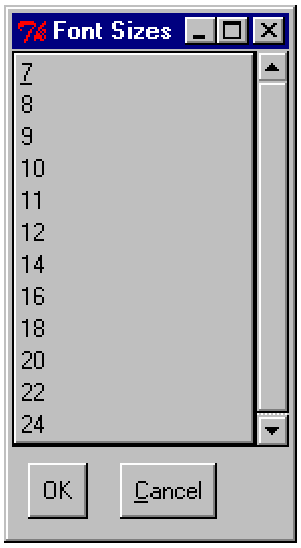


Figure 7-23. Font Sizes Selection Window

7.1.3.7 Help Menu

ICTS provides context-sensitive help through the F1 key and the Help drop-down menu. Shows the Help drop-down menu:

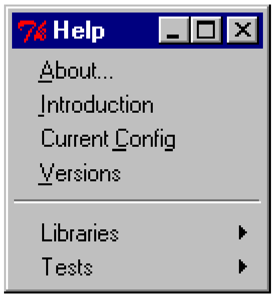


Figure 7-24. Help Menu

The Help menu contains these items:

- About. . .—Presents a window displaying the program name, version number and copyright information.
- Introduction—Displays a brief how-to-get-started tutorial in the message window. The tutorial appears automatically if the framework starts without a host configuration file.
- Current Config—Displays the current value of all host, platform, target, and user configuration variables in the message window.
- Versions– Displays version number information for various components of the framework in the message window.

- Libraries—Displays a cascading list of loaded libraries. Selecting an item from the list displays help for that item. The list contents depend on the libraries currently loaded.
- Tests—A cascading menu containing an entry for each loaded test module. Selecting an entry displays information about the corresponding test in the message window. Intel® Network Interface Card (LAN)

8 Running Commnad Compliance Report Test

8.1 Introduction

Command Compliance Report provides a summary of command support information on the target along with the information if a command is optional or mandatory according to IPMI1.0 / 1.5/2.0 specification. The scope of CCR is to only test the command support information and not the functionality of commands as defined under the IPMI1.0 / 1.5/2.0 Specifications.

CCR serves as a quick starting point in the IPMI Compliance testing, where this primary reports bring out the inconsistencies with the standard in terms of a list of Mandatory commands that are not supported on the target under test.

8.2 Scope of Compliance Test

- ICTS generates one CCR for each available interface.
- If the interface used for generation is a session based interface, then three separate CCRs are generated for each privilege level viz. Administrator, Operator and User
- Each CCR covers all the micro-controllers available on the target under test, except in case of session based interfaces, where it is generated only for BMC.
- CCR collects only the support information of the commands for a given Microcontroller and provides information if a command is mandatory or optional as per the IPMI 1.0 / 1.5/2.0 specifications
- While deciding if a given command is mandatory or optional, CCR considers the dependencies as defined in the IPMI1.0 / 1.5/2.0 standard.

NOTE

The dependency for Command ‘Get BT Interface Capability’ is presently not implemented in ICTS CCR test script, hence will always be printed as unknown

8.3 Report Format

Figure 8-1 shows an example CCR generated on LAN interface

Header Information:

Platform: Platform name as defined in the ICTS Platform file i_platform.tcl

Firmware Revision: Firmware Revision for each controller on the Target Server.

Interface: The interface on which the CCR has been generated

Level: Privilege Level at which the commands are sent. This is present only when the interface used is a session based interface like LAN Serial Interface

Time Stamp: Day – Date – Time on which the CCR was generated.

Command Support Information

In this area there are various columns for each command sent as described below

Command Name: Abbreviated Command Name

Micro Name: Hex Address of the Microcontroller over which the command was sent

Net Function: The Command Net Function as in IPMI specifications

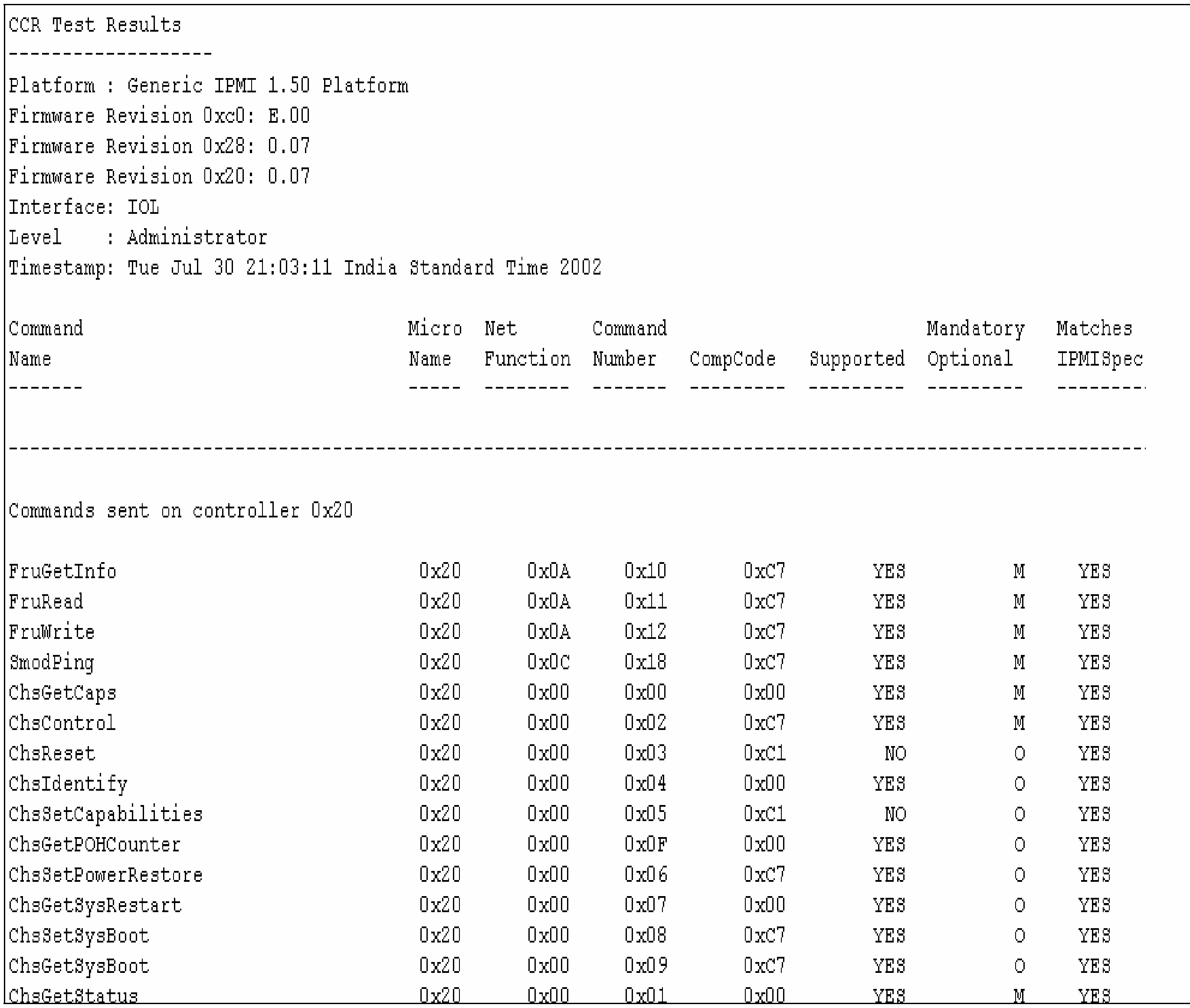
Command Number: The Command Number in Hex as defined in IPMI Specifications

Comp Code: Command Completion Code

Supported: Command is supported or not

Mandatory / Optional: If the given command is Mandatory or Optional on the platform under test considering the dependencies defined in IPMI Specifications. These are summarized in details in Section 4 of this Document.

Matches IPMI Specs: This column shows if the command support information is in accordance of IPMI 1.0 / 1.5/2.0 Specifications. Generally for the commands which were found Mandatory but not supported, this column will display a ‘No’ (Mismatch) with the IPMI Specifications.

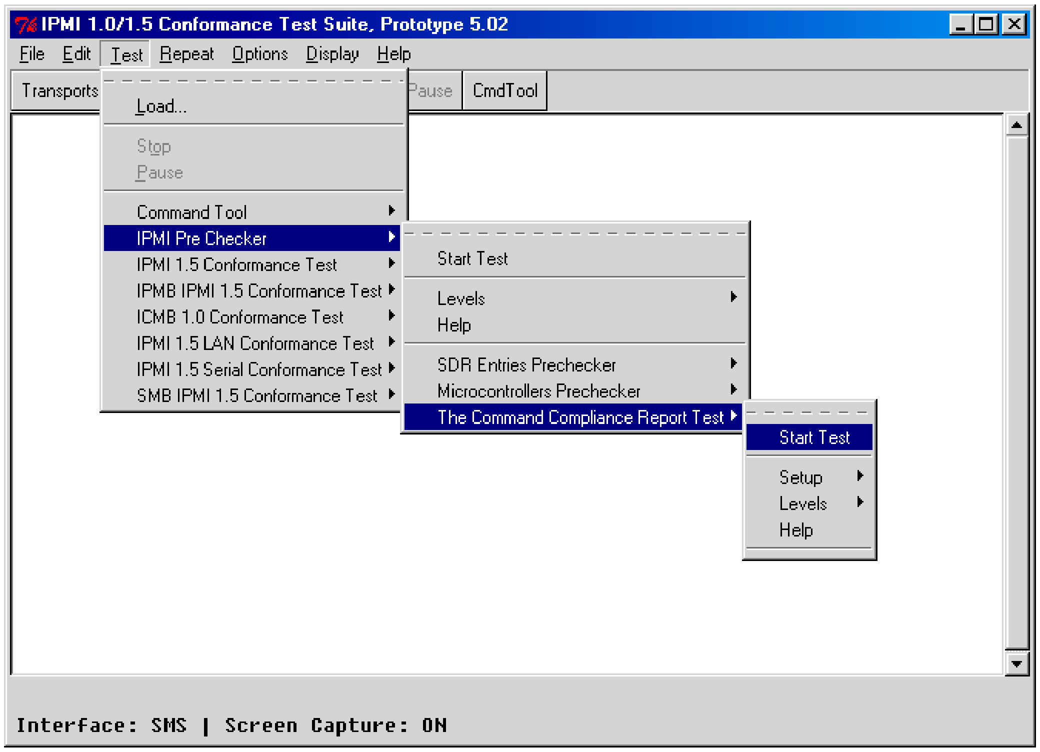


```
CCR Test Results
-------------------
Platform : Generic IPMI 1.50 Platform
Firmware Revision 0xc0: E.00
Firmware Revision 0x28: 0.07
Firmware Revision 0x20: 0.07
Interface: IOL
Level    : Administrator
Timestamp: Tue Jul 30 21:03:11 India Standard Time 2002

Command                              Micro  Net       Command                          Mandatory   Matches
Name                                 Name   Function  Number   CompCode   Supported  Optional    IPMISpec
-------                              -----  --------  -------  ---------  ---------  ---------   --------


-------------------------------------------------------------------------------------------------------


Commands sent on controller 0x20

FruGetInfo                            0x20      0x0A     0x10       0xC7        YES         M     YES
FruRead                               0x20      0x0A     0x11       0xC7        YES         M     YES
FruWrite                              0x20      0x0A     0x12       0xC7        YES         M     YES
SmodPing                              0x20      0x0C     0x18       0xC7        YES         M     YES
ChsGetCaps                            0x20      0x00     0x00       0x00        YES         M     YES
ChsControl                            0x20      0x00     0x02       0xC7        YES         M     YES
ChsReset                              0x20      0x00     0x03       0xC1         NO         O     YES
ChsIdentify                           0x20      0x00     0x04       0x00        YES         O     YES
ChsSetCapabilities                    0x20      0x00     0x05       0xC1         NO         O     YES
ChsGetPOHCounter                      0x20      0x00     0x0F       0x00        YES         O     YES
ChsSetPowerRestore                    0x20      0x00     0x06       0xC7        YES         O     YES
ChsGetSysRestart                      0x20      0x00     0x07       0x00        YES         O     YES
ChsSetSysBoot                         0x20      0x00     0x08       0xC7        YES         O     YES
ChsGetSysBoot                         0x20      0x00     0x09       0xC7        YES         O     YES
ChsGetStatus                          0x20      0x00     0x01       0x00        YES         M     YES
```

Figure 8-1. CCR Example

8.4 Running the ICTS CCR Test

To generate the CCR, the CCR test needs to be run. Before running this test SMS or I2C interface must be available. The test is run from the ICTS Pre-checker menu as shown in Figure 8-2


IPMI 1.0/1.5 Conformance Test Suite, Prototype 5.02
File Edit Test Repeat Options Display Help
Transports
Pause
CmdTool
Load...
Stop
Pause
Command Tool
IPMI Pre Checker
IPMI 1.5 Conformance Test
IPMB IPMI 1.5 Conformance Test
ICMB 1.0 Conformance Test
IPMI 1.5 LAN Conformance Test
IPMI 1.5 Serial Conformance Test
SMB IPMI 1.5 Conformance Test
Start Test
Levels
Help
SDR Entries Prechecker
Microcontrollers Prechecker
The Command Compliance Report Test
Start Test
Setup
Levels
Help

Interface: SMS | Screen Capture: ON

Figure 8-2. Running CCR Test

8.5 CCR naming convention

The test also manages the amount of disc-space used by the reports in ICTS Logs folder. If a report for a given interface (and privilege level for session based interface) already exists, the CCR test renames the extension of the existing report to .old from .txt, and then generates the new report with the same name but with extension .txt. Thus the number of reports for the same interface and/or privilege level could not exceed 2. This ensures the disc space is not flooded by the reports if the test in run multiple times.

The Report Names are generated with following convention

<Platform name as in i_platform.tcl> platform_<transport name>_<privilege level>.txt

Note: Privilege level is A/O/U for Administrator/Operator/User and is present only for session based interfaces

8.6 Interpretion of result

In CCR the “Matches IPMI Specs” column can indicate three different values significance of which is explained below

- Yes: This means the command support is as specified in the IPMI Specifications.
- No: This means the command support is not as specified in the IPMI Specifications. The general case would be where a command which is mandatory in IPMI specifications is not supported on the platform under test.

Note1: This indicates a Firmware Error, where the command is supported on a privilege level lower than the required privilege as mentioned in IPMI 1.5/2.0 Specifications Appendix G. This error is applicable for only IPMI1.5/2.0 Targets

9 Functional Conformance Report

9.1 Introduction

Functional Conformance Report (FCR) is an extended feature to ICTS (IPMI conformance test suite). FCR reports each test case is loaded or not, executed or not and the executed result under each applicable execute mode during the tracking time span. Also statistical data is provided to show a whole picture of the test result. FCR could be generated on demand of user during ICTS run or automatically before ICTS exit.

9.2 Report Format

FCR reports each test case is loaded or not, executed or not and executed result under each applicable execute mode during the tracking time span.

The executed result defined as:

P	– Pass
F	– Fail
I/F	– Init Fail
A	– Abort (the test dependency is not available)
---	– Not Applicable

The execute mode defined as:

SMS	– Execute the test case through system management interface (KCS, SMIC, BT).
I2C	– Execute the test case through IPMB interface
SMB	– Execute the test case through SMBus interface
IOS	– Execute the test case through Serial Modem interface
	(Basic Mode, Terminal Mode, PPP/UDP Mode)
IOL	– Execute the test case through LAN interface
ICMB	– Execute the test case through ICMB interface

The time span is defined as:

From	– Start of ICTS
To	– The demand of FCR generate by user OR exit of ICTS

Statistical data is provided in report. They are:

Number of Test cases loaded

Number of Test cases executed

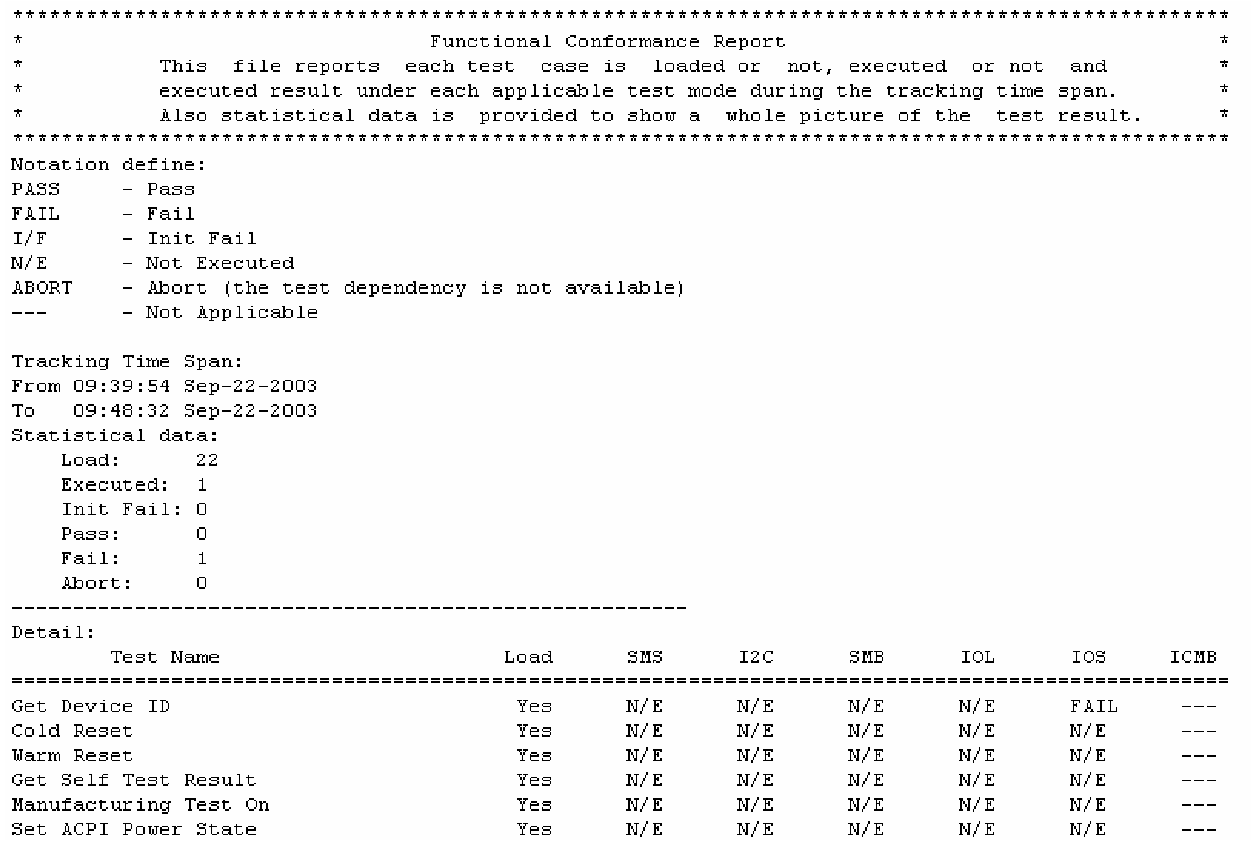
Number of Test cases passed

Number of Test cases failed

Number of Test cases aborted

Number of Test cases initialize failed

Figure 9-1 is a screen shot of an FCR report



```
************************************************************************************************
*                                Functional Conformance Report                                 *
*           This  file reports  each test  case is  loaded or  not, executed  or not  and       *
*           executed result under each applicable test mode during the tracking time span.      *
*           Also statistical data is  provided to show a  whole picture of the  test result.    *
************************************************************************************************
Notation define:
PASS     - Pass
FAIL     - Fail
I/F      - Init Fail
N/E      - Not Executed
ABORT    - Abort (the test dependency is not available)
---      - Not Applicable

Tracking Time Span:
From 09:39:54 Sep-22-2003
To   09:48:32 Sep-22-2003
Statistical data:
    Load:      22
    Executed:  1
    Init Fail: 0
    Pass:      0
    Fail:      1
    Abort:     0
-------------------------------------------------------
Detail:
        Test Name                      Load      SMS      I2C      SMB      IOL      IOS      ICMB
================================================================================================
Get Device ID                           Yes      N/E      N/E      N/E      N/E      FAIL      ---
Cold Reset                              Yes      N/E      N/E      N/E      N/E      N/E       ---
Warm Reset                              Yes      N/E      N/E      N/E      N/E      N/E       ---
Get Self Test Result                    Yes      N/E      N/E      N/E      N/E      N/E       ---
Manufacturing Test On                   Yes      N/E      N/E      N/E      N/E      N/E       ---
Set ACPI Power State                    Yes      N/E      N/E      N/E      N/E      N/E       ---
```

Figure 9-1 FCR Report

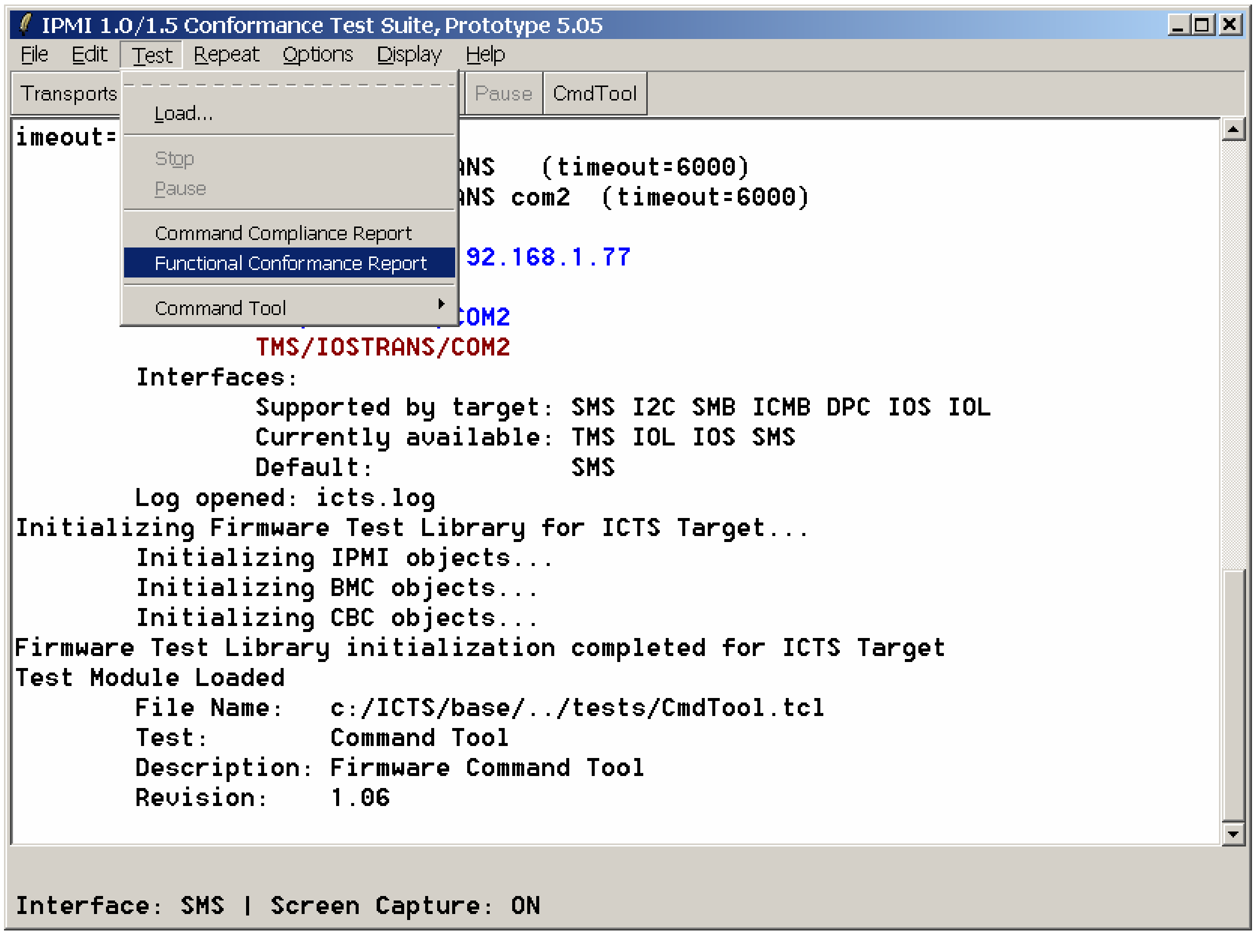
9.3 Generate FCR report

FCR report is gernearted from Test Menu as Figure 9-2 show, or will be generate when exit ICTS automatically. Generate FCR automatically when ICTS exit feature can be disabled by modified the user configuraion file ([ICTS_ROOT]\users\i_user.tcl) FCR generation section:

# Functional Report Auto Generate

# 1 – enable, 0 - disable

set FCR_Generate 1


IPMI 1.0/1.5 Conformance Test Suite, Prototype 5.05
File Edit Test Repeat Options Display Help
Transports Pause CmdTool
Load...
Stop
Pause
Command Compliance Report
Functional Conformance Report
Command Tool

```
imeout=
                                 NS   (timeout:6000)
                                 NS com2  (timeout:6000)

                                 92.168.1.77

                                 COM2
                TMS/IOSTRANS/COM2
        Interfaces:
                Supported by target: SMS I2C SMB ICMB DPC IOS IOL
                Currently available: TMS IOL IOS SMS
                Default:             SMS
        Log opened: icts.log
Initializing Firmware Test Library for ICTS Target...
        Initializing IPMI objects...
        Initializing BMC objects...
        Initializing CBC objects...
Firmware Test Library initialization completed for ICTS Target
Test Module Loaded
        File Name:   c:/ICTS/base/../tests/CmdTool.tcl
        Test:        Command Tool
        Description: Firmware Command Tool
        Revision:    1.06

Interface: SMS | Screen Capture: ON
```

Figure 9-2 Generate FCR report

9.4 FCR naming convention

The Report Names are generated defaultly with following convention

<Time>@<MON_DD_YYYY>.fcr

Also user can specify the report name and save path if generate FCR report by Test -> Functional Conformance Report.

10 Interface Conformance Test

The interface conformance test module, versus command comformance test, is intended to verify whether or not a target platform's interface implementation is conformant to IPMI 1.5/2.0. This test module consist of six groups of test cases corresponds to six IPMI interfaces defined in IPMI 1.5/2.0 as follow:

SMS interface

- KCS interface
- SMIC interface
- BT interface

SMBus interface

IPMB interface

LAN interface

Serial/Modem interface

- Basic Mode
- Terminal Mode
- PPP/UDP Mode

ICMB interface

Following sections will describe how to use these test groups to verify the conformability of the IPMI interface implementation.

10.1 SMS Interface

Local software running on the managed system and using the System Interface to the BMC will generally be referred to as *system manangement software* or SMS interface. IPMI 1.5/2.0 specified three supported BMC to SMS interface: KCS, SMIC and BT interface.

10.1.1 Keyboard Controller Style (KCS) interface

10.1.1.1 Stragey & Configuration

The test cases are implemented in two parts: first are the test routines in the SIOPROXY.EXE that run on target platform. Second are the TCL based test triggers that will trig these test routines by sending routine trig message and analysis response message from test routines to determine the test result. The trig message and response message will be sent and received using Message Routing Library, routing through SIO transport. So the KCS interface test configuration will be as showed in Figure 10-1

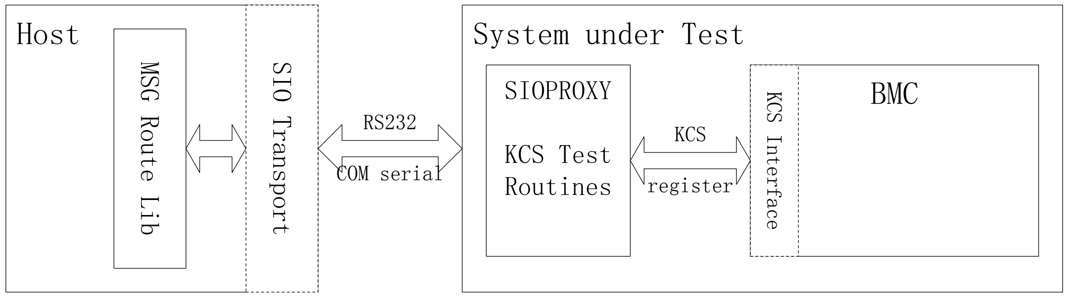


Figure 10-1. KCS interface test configuration

10.1.1.2 Software & Hardware setup

10.1.1.2.1 Hardware setup

1. Null modem connection serial port of test host and test target

10.1.1.2.2 Software setup

1. Test host run ICTS with environment setup as:

   set ICTSTRANS=SIOTRANS

   set ICTSCOM=<serial port number> // null modem connected with test target

2. Test target booting to MS-DOS 6.0 and start sioproxy.exe with

   Sioproxy –c# // # is the serial port number null modem connected with test host

3. Test host load KCS interface test cases in

   <ICTS_ROOT>\ tests\ICTS_tests\itf_tests\KCS

10.1.2 Server Management Interface Chip (SMIC) Interface

This interface test use the same test stragey configuration as KCS interface, so the software and hardware setup should be the same as KCS, except that we should load test cases in

<ICTS_ROOT>\ tests\ICTS_tests\itf_tests\SMIC

10.1.3 Block Transfer (BT) Interface

This interface test use the same test stragey configuration as KCS interface, so the software and hardware setup should be the same as KCS, except that we should load test cases in

<ICTS_ROOT>\ tests\ICTS_tests\itf_tests\BT

10.2 SMBus Interface

10.2.1 SMBust Interface test strategy & configuration

As *IPMI version 1.5/2.0 revision 1.1* (page 42, Table 6-2 & related notes) specified, SMBus interface talk with the other devices connected on the SMBus using IPMI-SMBus protocol. IPMI-SMBus protocol describes the format of the IPMI message which will be carried on SMBus. The IPMI-SMBus message format is designed intentionally illegal with respect to the SMBus specification protocols (the 9 bus protocols), in order to provide a way for a management controller to unambiguously differentiate IPMI messages from SMBus transactions.

To test the SMBus interface, we should have the device that adopts the SMBus physical layer and data-link layer specification. A saelig card with the transport module is the good choice. The IPMI-on-SMBus protocol architecture and test configuration is showed in Figure 10-2.

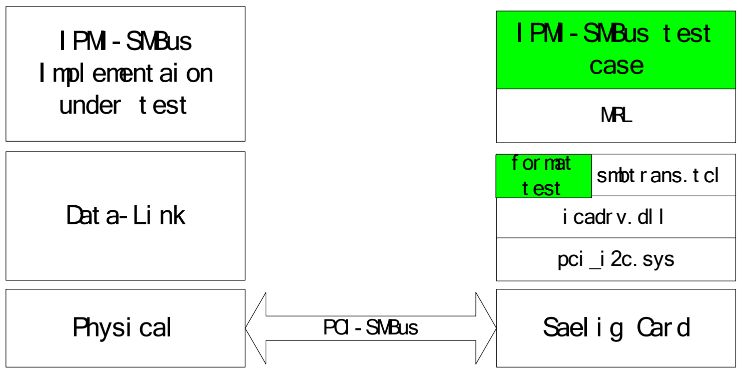


Figure 10-2 IPMI-on-SMBus protocol architecture & test configuration

As Figure 10-2 shows, the IPMI-SMBus test cases will use the services provided by the low-level stuff to send and receive the IPMI-on-SMBus request and response messages. What the test case cared and verified is the protocol message format, including:

1. Address
2. Net function code
3. 00h field
4. Package Error Checking
5. Sequence

10.2.2 Software and Hardware setup

10.2.2.1 Hardware setup

1. Saelig card installed on test host
2. Connect Saling card with PCI-SMBus on test target

10.2.2.2 Software setup

1. Saelig card driver installed
2. Run ICTS with environment set as follow:

   set SMBTRANS=SMBTRANS

   set SMBPORT=0

3. Load test cases in

   <ICTS_ROOT>\ tests\ICTS_tests\itf_tests\smbus

10.3 IPMB Interface

IPMB interface test using the same test stragey and configuration as SMBus, so the hardware and software setup of IPMB interface test is the same as SMBus interface test except the test cases should load from:

<ICTS_ROOT>\ tests\ICTS_tests\InterfaceSpecific\ipmb

10.4 LAN Interface

Lan interface test will test the LAN Interface specified by IPMI specification version 1.5/2.0. LAN interface is used to transfer IPMI message between BMC and a remote management system over an Ethernet LAN connection using UDP under Ipv4. The UDP datagrams are formatted to contain IPMI request and response messages, plus messages for discovery and authentication.

LAN interface conformance requirements:

1. Support ASF ping/pong message for system discovery
2. IPMI message format must conformance to specification section 12.4
3. LAN session establish procedure

10.4.1 LAN interface conformance test configuration

No special configuration for LAN conformance test configuration.

10.4.2 Software & Hardware setup

Connect NIC on test target, which side band or dedicated to BMC, to LAN.

Setting environment as:

set IOLTRANS=IOLTRANS

set IOLPORT= <ip address setting for LAN interface>

set IOLOPT=-privlev <priv> -authtype <authtype> -timeout 1000 -retries 3 -user <username> -password <password>

10.5 Serial/Modem Interface

10.5.1 Basic Mode

The following behavior will be tested:

1) Connection Mode Auto-detect
2) Active Ping Message
3) Callback
4) Basic Mode IPMI messaging (Packet Framing, Data Byte Escaping, Message retries, Packet Handshake)
5) Serial Port switch
6) Session

The following configuration will be used to executing Basic Mode SoM tests:

Null Modem cable will be used to connect Test Host and test target. IPMB and SMS interface should be available for some of the test cases.

10.5.2 Terminal Mode

The following behavior will be tested:

1) Terminal Session
2) Query System Health
3) Power On/Off
4) Enable/Disable Line Editing
5) Del Control in Line Editing
6) Input Restrictions
7) New Line Sequence

A direct or modem connection from Host to Target, IPMB or SMS interface should be available to run Terminal Mode test cases.

10.5.3 PPP/UDP Mode

This mode of operation uses PPP [RFC1661] (point-to-point protocol) messaging for transmitting IP packets on an asynchronous link per [RFC1662]

PPP/UDP mode transfers IPMI Messages encapsulated in RMCP Packets. This enables RMCP ASF Messages as well as IPMI Messages to be delivered to the BMC. The RMCP Packets are carried within UDP datagram using the same format as the IPMI LAN messages. The resultant UDP datagram are transferred within PPP frames.

10.5.3.1 Test stragey & Configuration

PPP/UDP mode tests call Windows RAS dial library, establish a PPP connection to Target. Then the Host can establish a PPP/UDP Session over IOLTRANS. PPP/UDP mode use IOLTRANS as its transport.

10.5.3.2 Software & Hardware setup

PPP/UDP mode tests require a direct serial connection to target's Serial/Modem port. The BMC acts as a PPP dial in server. The Host's should be running Windows 2000 or XP.

Startup batch file

A SIO or I2C transport is required to setup PPP/UDP parameters; I2C transport has a high priority than SIO.

IOS transport should be configured in startup batch file.

IOLTRANS should be enabled, because PPP/UDP connection require IOL transport library.

Set the IOLPORT to any IP address is unconcerned, when IOL session initialization failed, it's not error, just ignore it.

Example of variable settings in startup batch file:

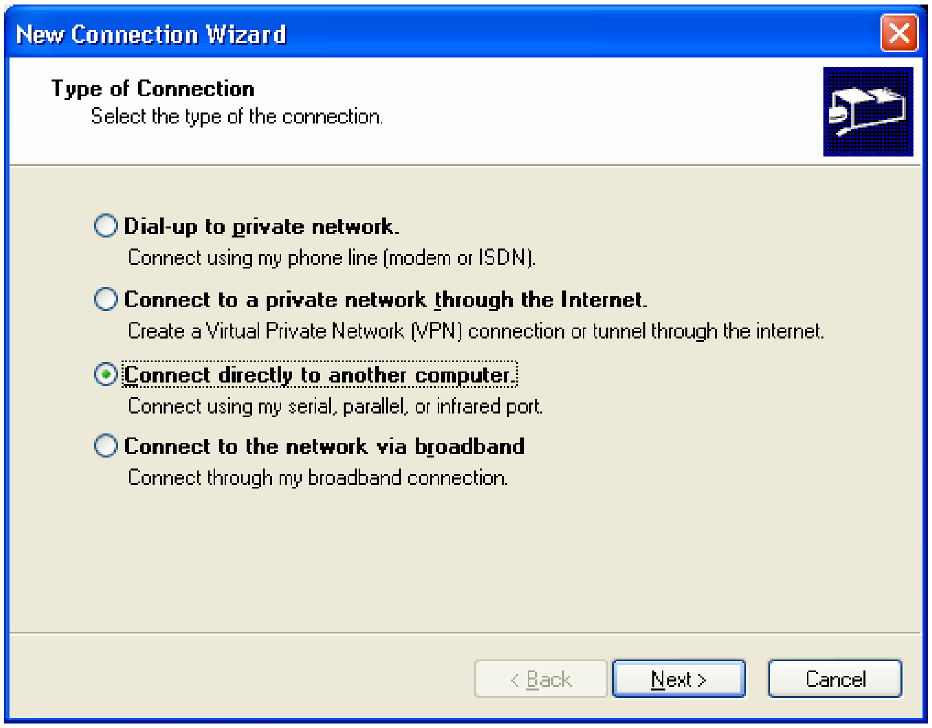
```
set ICTSTRANS=SIOTRANS
set ICTSCOM=COM1
set ICTSMODE=DEVELOP

set IOSTRANS=IOSTRANS
set IOSCOM=COM2

set IOLTRANS=IOLTRANS
set IOLPORT=192.168.1.46
set IOLOPT=-priv admin -auth md2 -timeout 3000 -retries 3
```

10.5.3.3 Create Null Modem connection

When the PPP/UDP cases are being loaded, the setup routine will check or create a null modem connection named "ICTS". ICTS use this connection to dial BMC's PPP/UDP interface.

The user running the test must have rights to create new connection, or the setup will fail.

During the new connection wizard, choose direct connect to another computer via COM2, and set maximum speed to 19200 bps, no flow control. As following screen shots show:



New Connection Wizard

Type of Connection

Select the type of the connection.

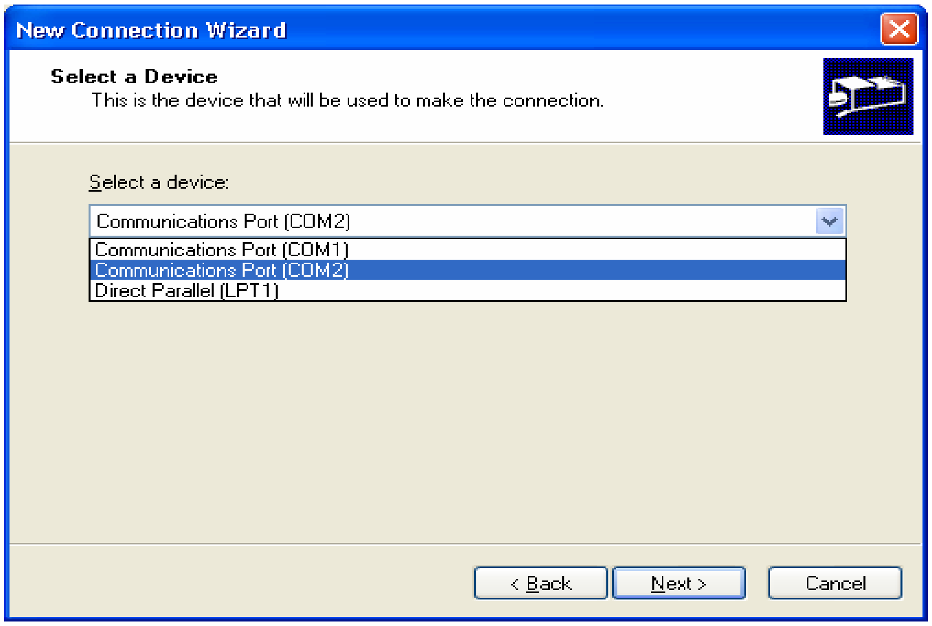
( ) **Dial-up to private network.**
Connect using my phone line (modem or ISDN).

( ) **Connect to a private network through the Internet.**
Create a Virtual Private Network (VPN) connection or tunnel through the internet.

(●) **Connect directly to another computer.**
Connect using my serial, parallel, or infrared port.

( ) **Connect to the network via broadband**
Connect through my broadband connection.

Figure 10-3 New Connection Wizard



New Connection Wizard

Select a Device

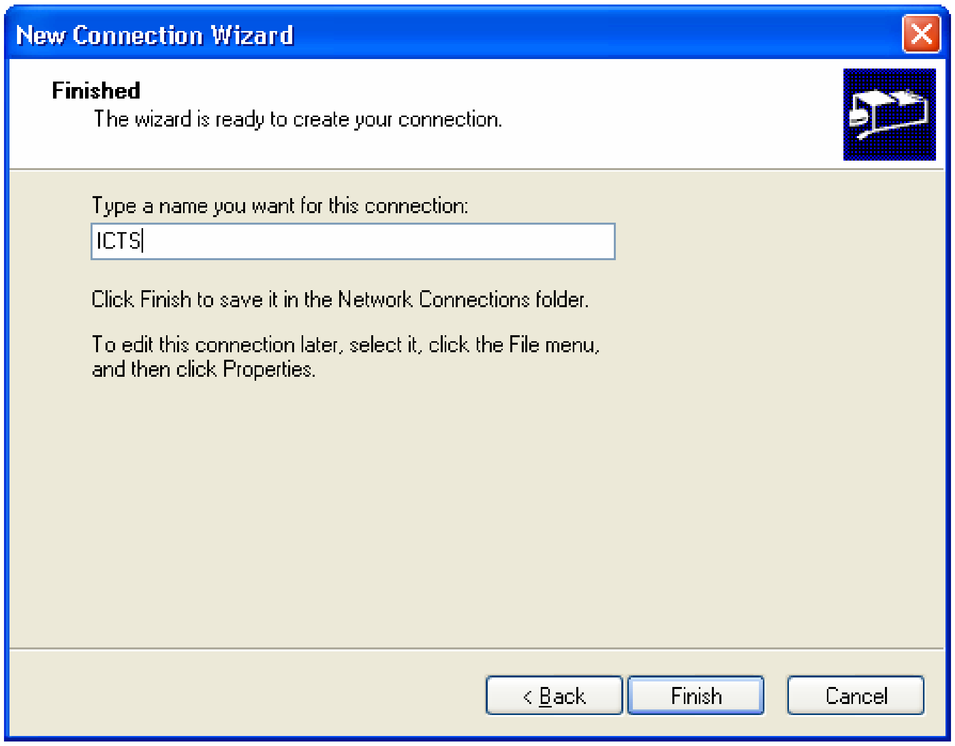
This is the device that will be used to make the connection.

Select a device:

Communications Port (COM2)

- Communications Port (COM1)
- Communications Port (COM2)
- Direct Parallel (LPT1)

Figure 10-4 Select Com Port


New Connection Wizard

Finished

The wizard is ready to create your connection.

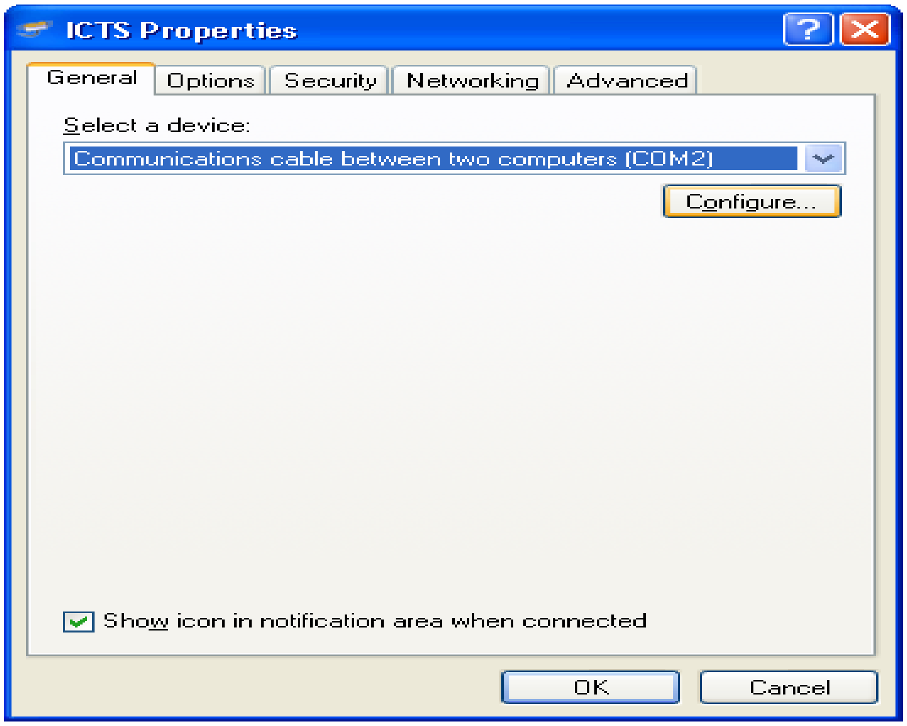
Type a name you want for this connection:

ICTS

Click Finish to save it in the Network Connections folder.

To edit this connection later, select it, click the File menu, and then click Properties.

Figure 10-5 Accept connect name

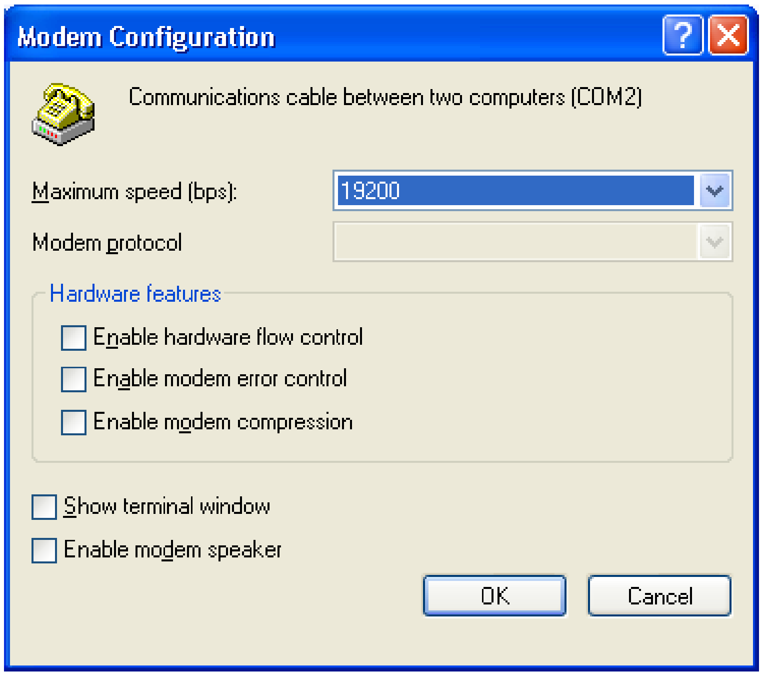

ICTS Properties

Select a device:

Communications cable between two computers (COM2)

✓ Show icon in notification area when connected

Figure 10-6 Choose Modem


Modem Configuration

Communications cable between two computers (COM2)

Maximum speed (bps): 19200

Modem protocol

Hardware features

- [ ] Enable hardware flow control
- [ ] Enable modem error control
- [ ] Enable modem compression

- [ ] Show terminal window
- [ ] Enable modem speaker

Figure 10-7 Configure Modem

10.6 ICMB Interface

ICMB interface is the ICMB bridge controller that connected to an existing IPMI implementation that contained an IPMB. The ICMB bridge provides a mechanism to allow a system to access a remote system's IPMB. The ICMB specification version 1.0 revision 1.2 defines the ICMB bus and ICMB bridge.

ICMB datalink conformance test will verify whether the implementation of datalink layer of the ICMB Bridge controller is conformance to the ICMB datalink protocol. As for the test of ICMB bridge function please refer ICTS (Test Case) IPS section16 ICMB TEST UPDATE. ICMB 1.0 specification for Datalink Protocol includes the following requirements relate to:

1. packet framing,
2. packet format (general, message bridge, event message)
3. packet checksum

For IPMI 1.5 specification, ICMB bridge commands could be implemented directly as BMC commands, and use IPMI 1.5 channels and "Send Message" command to replace the ICMB "bridge request" commands to bridge IPMI message from System Interface / Local IPMB to remote IPMB. When this way be chose to implement ICMB interface, the message bridge format for system interface / local IPMB to remote IPMB should be verified as IPMI specification 1.5 section 8.2.1 & section 8.2.2.

10.6.1 ICMB interface test strategy & configuration

ICMB packets consist of a series of eight bit bytes. The ICMB uses standard RS-232 UART character framing. Packets are framed in a way that allows the start and end of a packet to be easily distinguished. Certain characters have been chosen as framing characters and the protocol guarantees that those characters will not appear on the bus except to mark packet boundaries (excluding a HW or SW failure).

To test ICMB, we should have a reference target, which pure adopts ICMB 1.0 specification. An RS-485 card plus transport layer for this card is a good selection.

Also test cases will send command “Get Bridge Statistics” by system interface to get Host Bridge’s Datalink packet statistics to verify some of the ICMB bridge implementation requirements, so the SIOTRANS should be available for ICMB interface test. ICMB interface test configuration showed by as Figure 10-8 below:

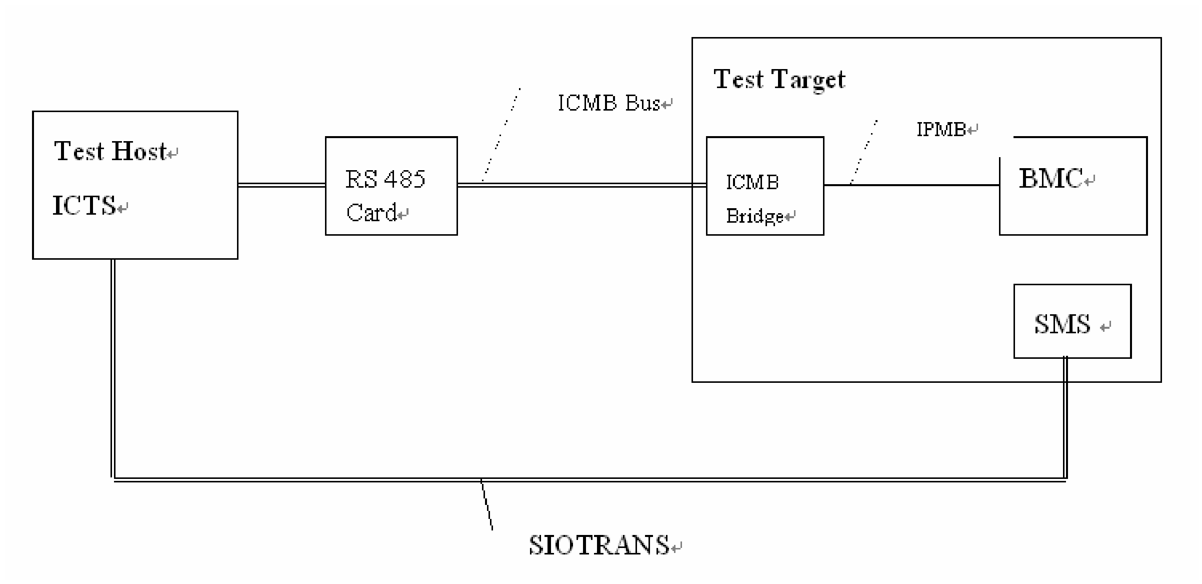


Figure 10-8 ICMB Test configuration

10.6.2 Hardware & Software setup

Hardware:

1. RS485 card connect to serial port of test host
2. Connection RS485 card to ICMB bridge connector of test target

Software:

Environment setup as follow on test host:

set ICMBTRANS=ICMBTRANS

set ICMBCOM= <serial port number>   // connected to RS485 card

Appendix

Null Modems

A Null Modem is used to connect two DTE together.

DTE is acronym for Data Terminal Equipment. Examples of DTE are computers, printers & terminals. DCE is acronym for Data Communication Equipment. An example of DCE is modems.

This is commonly used as a cheap to transfer files between computers using Zmodem Protocol, Xmodem Protocol etc. This can also be used with many Microprocessor Development Systems.

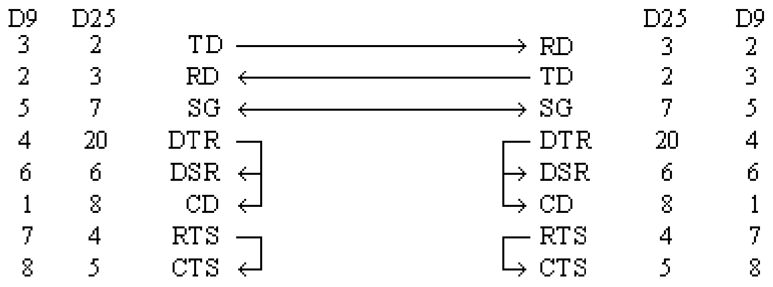


Figure 1: Null Modem Wiring Diagram

Note: DSR & CD are jumpered to fool the programs to think that they are online.

Above is a method of wiring a Null Modem. It only requires 3 wires (TD, RD & SG) to be wired straight through thus is more cost effective to use with long cable runs. The theory of operation is reasonably easy. The aim is to make to computer think it is talking to a modem rather than another computer. Any data transmitted from the first computer must be received by the second thus TD is connected to RD. The second computer must have the same set-up thus RD is connected to TD. Signal Ground (SG) must also be connected so both grounds are common to each computer.

The Data Terminal Ready is looped back to Data Set Ready and Carrier Detect on both computers. When the Data Terminal Ready is asserted active, then the Data Set Ready and Carrier Detect immediately become active. At this point the computer thinks the Virtual Modem to which it is connected is ready and has detected the carrier of the other modem.

All left to worry about now is the Request to Send and Clear To Send. As both computers communicate together at the same speed, flow control is not needed thus these two lines are also linked together on each computer. When the computer wishes to send data, it asserts the Request to Send high and as it's hooked together with the Clear to Send, It immediately gets a reply that it is ok to send and does so.

Notice that the ring indicator is not connected to anything of each end. This line is only used to tell the computer that there is a ringing signal on the phone line. As we don't have a modem connected to the phone line this is left disconnected.

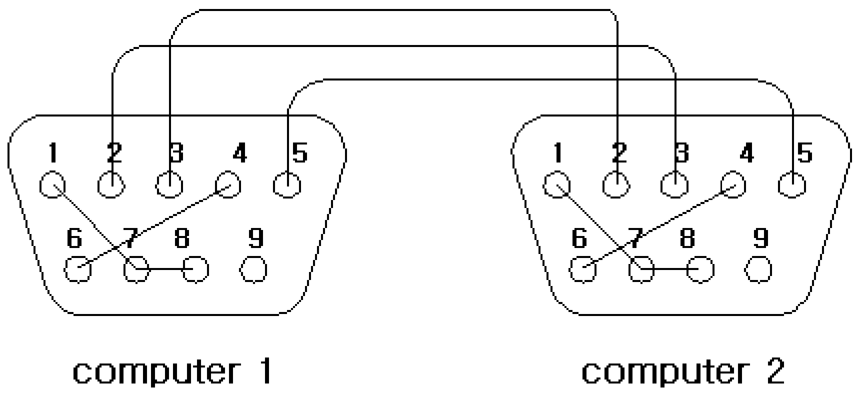


Figure 2: Pin connection between two DTE devices.

Abbreviation	Full Name	Function
TD	Transmit Data	Serial Data Output (TXD)
RD	Receive Data	Serial Data Input (RXD)
CTS	Clear to Send	This line indicates that the Modem is ready to exchange data.
DCD	Data Carrier Detect	When the modem detects a "Carrier" from the modem at the other end of the phone line, this Line becomes active.
DSR	Data Set Ready	This tells the UART that the modem is ready to establish a link.
DTR	Data Terminal Ready	This is the opposite to DSR. This tells the Modem that the UART is ready to link.
RTS	Request To Send	This line informs the Modem that the UART is ready to exchange data.
RI	Ring Indicator	Goes active when modem detects a ringing signal from the PSTN.

Index