Dialogic

Dialogic® System Release 6.1 CompactPCI for Windows®

Release Update

June 11, 2012

05-2514-057

Publication Date: June 11, 2012

Document Number: 05-2514-057

Dialogic® System Release 6.1 CompactPCI for Windows® Release Update

About This Publication

This section contains information about the following topics:

Purpose

This Release Update addresses issues associated with Dialogic® System Release 6.1 CompactPCI for Windows® (sometimes also referred to herein as "System Release 6.1 CompactPCI Windows"). In addition to summarizing issues that were known as of the Release's general availability, it is intended that this Release Update will continue to be updated to serve as the primary mechanism for communicating new issues, if any, that may arise after the release date.

Intended Audience

This Release Update is intended for users of System Release 6.1 CompactPCI Windows.

How to Use This Publication

This Release Update is organized into four sections (click the section name to jump to the corresponding section):

- Document Revision History: This section summarizes the ongoing changes and additions that are made to this Release Update after its original release. This section is organized by document revision and document section.
- Post-Release Developments: This section describes significant changes to the system release subsequent to the general availability release date. For example, the new features provided in Service Updates are described here.
- Release Issues: This section lists issues that may affect the system release hardware and software. The primary list is sorted by issue type, but alternate sorts by defect number, by product or component, and by Service Update number are also provided.
- Documentation Updates: This section contains corrections and other changes that apply to the System Release documentation set that were not made to the documents prior to the release. The updates are organized by documentation category and by individual document.

Related Information

See the following for additional information:

- For information about the products and features supported in this release, see the *Dialogic® System Release 6.1 CompactPCI for Windows® Release Guide*, which is included as part of the documentation bookshelf for the release.
- For further information on issues that have an associated defect number, you may use the Defect Tracking tool at *http://membersresource.dialogic.com/defects/*. When you select this link, you will be asked to either LOGIN or JOIN.
- *http://www.dialogic.com/support/* (for Dialogic technical support)
- *http://www.dialogic.com/* (for Dialogic® product information)

Document Revision History

This Revision History summarizes the changes made in each published version of the Release Update for Dialogic® System Release 6.1 CompactPCI for Windows®, which is a document that has been and is intended to be periodically updated throughout the lifetime of the release.

Document Rev 57 - June 11, 2012

Updated for Service Update 201.

In the Post Release Developments section, added :

- PDK Support for Automatic Answer and Reject of Inbound Calls.

In the Release Issues section, added the following resolved problems: IPY00080914, IPY00080931, IPY00081776, IPY00082087, IPY00091941, IPY00092204, IPY00092250, IPY00092546, IPY00093410, IPY00093957, IPY00094517, IPY00098919, IPY00099067, IPY00099099, IPY00099200, IPY00099304, IPY00099630, IPY00099791, IPY00099876, IPY00099886, IPY00099941, IPY00100083, IPY00100303.

Document Rev 56 - October 31, 2011

Updated for Service Update 199.

In the Post Release Developments section, added Status Monitor Tool Support for 16 Span Boards.

In the Release Issues section, added the following resolved problems: IPY00091862, IPY00092283, IPY00092350, IPY00092493, IPY00092923, IPY00093022, IPY00093270, IPY00093627, IPY00093701, IPY00093771, IPY00093815, IPY00094129, IPY00094190, IPY00094251, IPY00094385.

Document Rev 55 - June 3, 2011

Updated for Service Update 198.

In the Post Release Developments section, added Improvement to Call Progress Analysis.

In the Release Issues section, added the following resolved problems: IPY00091022, IPY00092252, IPY00092647, IPY00092912, IPY00092990.

In the Documentation Updates section, added information about using Global Call SS7 and the SS7 Software Development Kit Package in the *Dialogic® Global Call SS7 Technology Guide*.

Document Rev 54 - November 19, 2010

Updated for Service Update 196.

In the Post Release Developments section, added Media Load Support for the Dialogic® DM/V1200A-4E1-cPCI Board.

In the Release Issues section, added the following resolved problems: IPY00092028, IPY00092115, IPY00092212.

Document Rev 53 - August 3, 2010

Updated for Service Update 195.

In the Post Release Developments section, updated the Media Load 2 table in Section 1.10, "New Media Loads for Dialogic® DM/V4800BC Media Boards Using Special Coders", on page 36. The FSK and TrueSpeech features have been removed and are no longer available. This change was necessary to resolve IPY00082064.

In the Release Issues section, added the following resolved problems: IPY00082064, IPY00091379, IPY00091543, IPY00090734, IPY00091224, IPY00091039, IPY00091410 IPY00091452, IPY00091490.

Document Rev 52 - December 17, 2009

Updated for Service Update 194.

In the Post Release Developments section, added support for Handling non-2xx Responses to T.38 Switch.

In the Release Issues section, added the following resolved problem: IPY00081853.

In the Documentation Updates section, added information about T.38 Fax Servers and IP Call Scenarios to the Dialogic® Global Call IP Technology Guide.

Document Rev 51 - October 30, 2009

Updated for Service Update 193.

In the Release Issues section, added the following resolved problems: IPY00080011, IPY00081301.

In the Documentation Updates section, added a new error code in the *Dialogic® Global Call IP Technology Guide*.

Document Rev 50 - October 2, 2009

Updated for Service Update 192.

In the Post Release Developments section, added an Important Notice about System Release Update Installation.

In the Release Issues section, added the following resolved problems: IPY00080772, IPY00080944, IPY00081061.

In the Documentation Updates section, added a note in the *Dialogic® System Release 6.1 CompactPCI for Windows® Release Guide* that the update install should not be used when upgrading from a Service Update prior to SU 176 to a more recent build. (IPY00081147)

Document Rev 49 - published July 17, 2009

Updated for Service Update 191.

In the Release Issues section, added the following resolved problems: IPY00080252, IPY00080340, IPY00080516.

Document Rev 48 - published June 12, 2009

Updated for Service Update 190.

In the Release Issues section, added the following resolved problems: IPY00079561, IPY00079825, IPY00079866, IPY00080009, IPY00080020, IPY00080145, IPY00080244.

Document Rev 47 - published April 17, 2009

Updated for Service Update 189.

In the Release Issues section, added the following resolved problems: IPY00045524, IPY00079108, IPY00079353, IPY00079523, IPY00079551, IPY00079590, IPY00079648, IPY00079651, IPY00079668, IPY00079678, IPY00079691, IPY00079703, IPY00079716.

Document Rev 46 - published March 6, 2009

Updated for Service Update 188.

In the Post-Release Developments section, added Media LAN Disconnection Alarm Notification for Dialogic® DM/IP Boards.

In the Release Issues section, added the following resolved problems: IPY00079393, IPY00079477.

In the Documentation Updates section, added documentation updates to the following documents because of a new feature in the Service Update: *Dialogic® Global Call IP Technology Guide*, *Dialogic® IP Media Library API Programming Guide*, and *Dialogic® IP Media Library API Library Reference*.

Document Rev 45 - published February 6, 2009

Updated for Service Update 186.

In the Post-Release Developments section, added the Performance Technologies CPC5505-B3M3H1 SBC under Support for Compute Platforms.

In the Release Issues section, added the following resolved problems: IPY00078576, IPY00079212, IPY00079365.

In the Documentation Updates section:

- Added an update to the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* to indicate that some coders are not supported on the Dialogic® DM/V4800BC Board with certain media loads.
- Added an update to the *Dialogic® Global Call Country Dependent Parameters (CDP) for PDK Protocols Configuration Guide* for a new parameter in the Brazil R2 Bidirectional protocol *pdk_br_r2_io.cdp* file.

Document Rev 44 - published January 23, 2009

Updated for Service Update 185.

In the Release Issues section, added the following resolved problems: IPY00078978, IPY00079095, IPY00079160, IPY00079251.

In the Documentation Updates section:

- Added an update to the *Dialogic® Global Call IP Technology Guide* to indicate that the INFO method is included as part of the Allow header in SIP messages by default.
- Deleted the corrections for the *Dialogic® Digital Network Interface Software Reference, Dialogic® Fax Software Reference*, and *Dialogic® Global Call ISDN Technology Guide*, because these corrections have been incorporated into updated documents that are now on the online documentation bookshelf.

Document Rev 43 - published December 5, 2008

Updated for Service Update 180.

In the Post-Release Developments section:

- Added information about Media Load 10F for DM/V4800BC Media Board under New Media Loads for Dialogic® DM/V2400A-cPCI and DM/V4800BC Media Boards.
- Added Support for SFTP in Dialogic® Global Call SS7 Call Control Library.

In the Documentation Updates section, deleted the corrections for the *Dialogic® Global Call SS7 Technology Guide*, because these corrections have been incorporated into an updated document that is now on the online documentation bookshelf.

Document Rev 42 - published November 25, 2008

Updated for Service Update 179.

In the Release Issues section:

- Added the following resolved problem: IPY00078854.
- Added the following known problem: IPY00079022.

Document Rev 41 - published November 11, 2008

Updated for Service Update 178.

In the Release Issues section, added the following resolved problems: IPY00045159, IPY00045292, IPY00045395, IPY00045456, IPY00078445, IPY00078519.

In the Documentation Updates section, added an update to the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* for mixing Clear Channel with ISDN, CAS, or R2MF protocols on the same board.

Document Rev 40 - published October 23, 2008

Updated for Service Update 176.

In the Release Issues section, added the following resolved problems: IPY00044730, IPY00045184, IPY00045224, IPY00045239, IPY00045277, IPY00045293, IPY00045304, IPY00045376, IPY00045388, IPY00045440, IPY00045442. Also added a resolved problem with Host Install (no defect number) regarding an error message that can occur during the installation of the Dialogic® System Release Software if the installation of the DetectorsProj service fails.

In the Documentation Updates section:

- Added an update to the *Dialogic® System Release 6.1 CompactPCI for Windows® Software Installation Guide* for an error message that can occur during the installation of the Dialogic® System Release Software if the installation of the DetectorsProj service fails.
- Added an update to the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* for the ISDN **ProtocolType** parameter (IPY00045267).
- Added an update to the *Dialogic® Continuous Speech Processing API Library Reference* for the **ec_reciottdata( )** and **ec_stream( )** functions.
- Added an update to the *Dialogic® Voice API Programming Guide*, Application Development Guidelines chapter, regarding continuous speech processing (CSP) resource sharing between multiple processes.
- Added an update to the *Dialogic® Voice API Library Reference* for the **dx_rec( )**, **dx_reciottdata( )**, **dx_recvox( )**, and **dx_recwav( )** functions.

Document Rev 39 - published September 3, 2008

Updated for Service Update 175.

In the Release Issues section, added the following resolved problems: IPY00044100, IPY00044425, IPY00045132.

Document Rev 38 - published August 26, 2008

Updated for Service Update 174.

In the Release Issues section, added the following resolved problems: IPY00043907, IPY00044132, IPY00044185, IPY00044215, IPY00044273, IPY00044363, IPY00044432, IPY00044544, IPY00044561, IPY00044614, IPY00044686, IPY00044699, IPY00044700, IPY00044713, IPY00044779, IPY00044811, IPY00044832, IPY00044932.

In the Documentation Updates section:

- Made a correction to the sample code for **gc_GetFrame( )** in the *Dialogic® Global Call API Library Reference*.
- Added an update to the *Dialogic® Voice API Library Reference* for the **dx_OpenStreamBuffer( )** function (IPY00044981).

Document Rev 37 - published July 8, 2008

Updated for Service Update 170.

In the Release Issues section, added the following resolved problems: IPY00043307, IPY00043801, IPY00044199, IPY00044200.

In the Documentation Updates section, added two updates to the *Dialogic® Global Call SS7 Technology Guide*, one about Bearer Independent Call Control (BICC) signaling protocol not supported, and one about using dual resilient SIU configurations.

Document Rev 36 - published June 11, 2008

Updated for Service Update 168.

In the Post-Release Developments section, added Media Load 1E under New Media Loads for Dialogic® DM/V4800BC Media Boards Using Special Coders.

In the Release Issues section:

- Added the following resolved problems: IPY00042860, IPY00043077, IPY00043701, IPY00043818.
- Added the following known problem: IPY00043963.

In the Documentation Updates section:

- Added an update to the *Dialogic® Global Call API Library Reference* for the **gc_util_insert_parm_val( )** function (IPY00043078).
- Added an update to the *Dialogic® Global Call SS7 Technology Guide* about opening trunk devices for SS7.
- Added that new versions of the *Dialogic® Modular Station Interface API Programming Guide* and *Dialogic® Standard Runtime Library API Library Reference* are now available on the online documentation bookshelf.

Document Rev 35 - published May 23, 2008

Updated for Service Update 166.

In the Release Issues section, added the following resolved problems: IPY00042866, IPY00043230, IPY00043267, IPY00043292, IPY00043545.

In the Documentation Updates section, added information about using remote DCM under *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* and *Dialogic® IPT Series on Windows® Configuration Guide*.

Document Rev 34 - published May 15, 2008

Updated for Service Update 165.

In the Release Issues section, added the following resolved problems: IPY00043240, IPY00043430, IPY00043432, IPY00043443.

In the Documentation Updates section, added that a new version of the *Dialogic® Native Configuration Manager API Programming Guide* is now available on the online documentation bookshelf.

Document Rev 33 - published April 30, 2008

Updated for Service Update 164.

In the Release Issues section, added the following resolved problems: IPY00042601, IPY00042609.

Document Rev 32 - published April 23, 2008

Updated for Service Update 163.

In the Post-Release Developments section, added Media Load 2 under New Media Loads for Dialogic® DM/V4800BC Media Boards Using Special Coders.

In the Documentation Updates section:

- Added procedures for checking the firmware version and upgrading the firmware for Dialogic® IPT Boards in the *Dialogic® System Release 6.1 CompactPCI for Windows® Administration Guide*.
- Added that a new version of the *Dialogic® Global Call Country Dependent Parameters (CDP) for PDK Protocols Configuration Guide* is now available on the online documentation bookshelf.

Document Rev 31 - published April 8, 2008

Updated for Service Update 162.

In the Post-Release Developments section, added New Media Loads for Dialogic® DM/V4800BC Media Boards Using Special Coders.

In the Release Issues section, added the following resolved problems: IPY00041808, IPY00042828, IPY00042845, IPY00042934.

Document Rev 30 - published March 24, 2008

Updated for Service Update 160.

In the Release Issues section, added the following resolved problems: IPY00042336, IPY00042464, IPY00042528, IPY00042584.

In the Documentation Updates section, added that a new version of the *Dialogic® Event Service API Library Reference* is now available on the online documentation bookshelf.

Document Rev 29 - published March 14, 2008

Updated for Service Update 159.

In the Release Issues section, added the following resolved problems: IPY00042204, IPY00042300.

In the Documentation Updates section, added that a new version of the *Dialogic® Event Service API Programming Guide* is now available on the online documentation bookshelf.

Document Rev 28 - published March 5, 2008

Updated for Service Update 158.

In the Post-Release Developments section:

- Added File Management Enhancements for DebugAngel Tool.
- Added File Management Enhancements for PDK Trace Tool.

In the Release Issues section:

- Added the following resolved problem: IPY00042168.
- Added the following known problem: IPY00042226.

In the Documentation Updates section, added updates to the *Dialogic® System Software Diagnostics Guide* because of new features in the Service Update.

Document Rev 27 - published February 19, 2008

Updated for Service Update 157.

In the Release Issues section, added the following resolved problems: IPY00041407, IPY00041580, IPY00041740, IPY00041855.

In the Documentation Updates section, deleted the corrections for the *Dialogic® Global Call API Library Reference*, because these corrections have been incorporated into an updated document that is now on the online documentation bookshelf.

Document Rev 26 - published January 30, 2008

Updated for Service Update 156.

In the Post-Release Developments section, deleted the detailed descriptions about some Dialogic® Global Call SS7 features that were previously included in this section, because this information has been incorporated into the updated *Dialogic® Global Call SS7 Technology Guide* that is now on the documentation bookshelf.

In the Release Issues section, added the following resolved problems: IPY00041280, IPY00041296.

In the Documentation Updates section, added documentation update to the *Dialogic® Voice API Library Reference* for the **dx_getdig( )** function (IPY00038453).

Document Rev 25 - published January 4, 2008

Updated for Service Update 154.

In the Release Issues section, added the following resolved problems: IPY00039334, IPY00040743, IPY00041079, IPY00041118, IPY00041300, IPY00041421.

Document Rev 24 - published December 10, 2007

Updated for Service Update 152.

In the Post-Release Developments section, added information about Media Load 9F-MC for DM/V4800BC Media Board under New Media Loads for Dialogic® DM/V2400A-cPCI and DM/V4800BC Media Boards.

In the Release Issues section, added the following resolved problems: IPY00039661, IPY00040536, IPY00040685, IPY00040832, IPY00041078, IPY00041209, IPY00041233.

In the Documentation Updates section:

- Added that new versions of the following documents are now available on the online documentation bookshelf: *Dialogic® Audio Conferencing API Programming Guide*, *Dialogic® Audio Conferencing API Library Reference*, *Dialogic® Continuous Speech Processing API Programming Guide*, *Dialogic® Continuous Speech Processing API Library Reference*, *Dialogic® Standard Runtime Library API Programming Guide*, and *Dialogic® Standard Runtime Library API Library Reference*.
- Added documentation updates to the *Dialogic® Fax Software Reference* for additional return values for **ATFX_RESLN( )** and other related changes (IPY00040796).
- Added documentation update to the *Dialogic® Global Call ISDN Technology Guide* for additional firmware-related cause values when using Dialogic® DM3 Boards (IPY00041046).
- Added documentation updates to the *Dialogic® Voice API Programming Guide* and *Dialogic® Voice API Library Reference* for functions that are no longer supported (**r2_creatfsig( )** and **r2_playbsig( )**).

Document Rev 23 - published October 12, 2007

Updated for Service Update 148.

In the Post-Release Developments section:

- Added Configuring SIP Stack Parameters with Global Call.
- Added Disabling Automatic re-INVITE Message when Switching between Fax and Audio.

In the Release Issues section, added the following resolved problems: IPY00038391, IPY00039476, IPY00039538, IPY00039707, IPY00039965, IPY00040179.

In the Documentation Updates section, added documentation updates to the *Dialogic® Global Call IP Technology Guide* because of new features in the Service Update.

Document Rev 22 - published September 28, 2007

Updated for Service Update 146.

In the Post-Release Developments section, added a code example to the IP Multicast Client Support section showing how to start a Multicast client session.

In the Release Issues section, added the following resolved problems: IPY00039401, IPY00039847.

In the Documentation Updates section, added IPPARM_TDMDET_CNG_ENABLE and IPPARM_TDMDET_CNG_DISABLE parameter IDs for IPSET_TDM_TONEDET under *Dialogic® Global Call IP Technology Guide* (IPY00040070).

Document Rev 21 - published September 7, 2007

Updated for Service Update 145.

In the Post-Release Developments section, added IP Multicast Client Support.

In the Release Issues section, added the following resolved problems: IPY00038981, IPY00039155, IPY00039412, IPY00039586. Also added a resolved problem (no defect number) regarding **ipm_GetLocalMediaInfo( )** returning an IP address of "0.0.0.0" after a hot swap.

In the Documentation Updates section, added documentation updates to the *Dialogic® IP Media Library API Library Reference* because of a new feature in the Service Update. Also made corrections to the **ipm_ModifyMedia( )** function.

Document Rev 20 - published August 21, 2007

Updated for Service Update 143.

In the Post-Release Developments section:

- Added Media Load 9F for DM/V4800BC Media Board under New Media Load for Dialogic® DM/V2400A-cPCI and DM/V4800BC Media Boards.
- Corrected the Support for Compute Platforms section (formerly titled "Support for Peripheral Hot Swap (PHS) on Additional Compute Platforms") and added the ADLINK cPCIS-3320/AC chassis. (Also deleted a documentation update for the *Dialogic® System Release 6.1 CompactPCI for Windows® Release Guide* regarding support for PHS on additional platforms, which was incorrect.)

In the Release Issues section:

- Added the following resolved problems: IPY00038190, IPY00038433, IPY00038494, IPY00038533, IPY00038545, IPY00038551, IPY00038708, IPY00038849, IPY00038979, IPY00038991, IPY00038998, IPY00039032, IPY00039068, IPY00039492. Also added IPY00037918 (resolved in Service Update 139).
- Eliminated the link to view issues sorted by PTR number. (PTR numbers have been superseded by defect numbers. The PTR numbers still appear in the Release Issues table for historical purposes, but a version of the table sorted by PTR number is no longer provided.)

In the Documentation Updates section:

- Added a documentation update to the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* because of a new feature in the Service Update.
- Deleted the corrections for the *Dialogic® Global Call Country Dependent Parameters (CDP) for PDK Protocols Configuration Guide* because these corrections have been incorporated into the updated document that is now on the documentation bookshelf.

Document Rev 19 - published July 12, 2007

Updated for Service Update 140.

In the Release Issues section, added the following resolved problems: IPY00038060, IPY00038240, IPY00038365, IPY00038572, IPY00038894.

Document Rev 18 - published June 29, 2007

Updated for Service Update 139.

In the Post-Release Developments section:

- Added Troubleshooting Information for RTF Logs.
- Added Remote Diagnostics Package.
- Added Enhanced Diagnostics Tools.
- Updated the New Dialogic® Diagnostics Management Console section to add more tools that can now be executed: Pstndiag and StatusMon.

In the Release Issues section, added the following resolved problems: IPY00033228, IPY00036855, IPY00037841, IPY00038074, IPY00038244, IPY00038280, IPY00038407, IPY00038435, IPY00038524, IPY00038611. Also added IPY00032797, IPY00037166, and IPY00037861 (resolved in Service Update 135).

In the Documentation Updates section:

- Added updates to the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* about NFAS D channel backup (DCBU) supported on 4ESS, 5ESS, and NI-2.
- Added an update to the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* about active talker and scaling in conferences.
- Added updates to the *Dialogic® System Software Diagnostics Guide* for the new diagnostics features in the Service Update.
- Added an update to the *Dialogic® Fax Software Reference* about the default fax font (IPY00037855).

Made global changes to reflect Dialogic brand.

Document Rev 17 - published June 7, 2007

Updated for Service Update 136.

In the Post-Release Developments section, added New QoS Alarms for RTCP and RTP Inactivity.

In the Documentation Updates section, added documentation updates to the following documents because of a new feature in the Service Update: *Dialogic® Global Call IP*

Technology Guide, *Dialogic® IP Media Library API Programming Guide*, *Dialogic® IP Media Library API Library Reference*.

Document Rev 16 - published May 29, 2007

Updated for Service Update 135.

In the Post-Release Developments section:

- Added New Operating System Support for Windows® Server 2003 SP2.
- Added Support for Peripheral Hot Swap (PHS) on Additional Compute Platforms (Advantech MIC-3081B with MIC-3369C SBC, and Diversified Technologies PlexSys-4 with CPB4612 SBC).

In the Release Issues section, added the following resolved problems: IPY00034857, IPY00036248, IPY00036865, IPY00036919, IPY00037183, IPY00037351, IPY00037372, IPY00037373, IPY00037396, IPY00037432, IPY00037507, IPY00037607, IPY00037632, IPY00037708, IPY00037767, IPY00037796, IPY00037817, IPY00037818. Also added IPY00036799 (resolved in Service Update 127).

Note: The fix for defect **IPY00037796** may have an impact on existing DM3 applications; refer to the defect description in the Release Issues section.

In the Documentation Updates section:

- Added documentation updates to the *Dialogic® System Release 6.1 CompactPCI for Windows® Release Guide* for Windows® Server 2003 SP2 support and for PHS support on additional compute platforms.
- Added an update to the Media Load table under *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide*.
- Added an update for the **gc_InitXfer( )** function under *Dialogic® Global Call API Library Reference* (IPY00038401).
- Added an update for the **dx_setevtmsk( )** function under *Dialogic® Voice API Library Reference* (IPY00038053).

Document Rev 15 - published March 16, 2007

Updated for Service Update 128.

In the Release Issues section, added the following resolved problems: IPY00037262, IPY00037356, IPY00037493.

Document Rev 14 - published March 13, 2007

Updated for Service Update 127.

In the Post-Release Developments section:

- Added New Parameter for Adjusting Silence Threshold on DM3 Boards.

- Added File Management Enhancements for ISDNtrace Tool.
- Added New Media Load for DM/V2400A-cPCI Boards.

In the Release Issues section, added the following resolved problems: IPY00006707 (PTR 33803), IPY00007470 (PTR 32437), IPY00009499 (PTR 33932), IPY00028633 (PTR 35748), IPY00036347, IPY00036423, IPY00036469, IPY00036504, IPY00036861, IPY00037004.

In the Documentation Updates section:

- Added documentation updates to the following documents because of new features in the Service Update: *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide*, *Dialogic® System Software Diagnostics Guide*.
- Added information about binary log files to the *Dialogic® System Software Diagnostics Guide* (IPY00037518).

Document Rev 13 - published February 20, 2007

Updated for Service Update 125.

In the Post-Release Developments section:

- Added Modified Alarm Events for Media LAN Disconnect.
- Added Enhancement to its_sysinfo Tool.
- Updated the New Diagnostics Management Console section to add more tools that can now be executed by it (these tools were previously available, but could only be executed independently before): AppMon, Castrace, Isdntrace, Dlgsnapshot, Dm3post, Debugangel, and Pdktrace.

In the Release Issues section:

- Added the following resolved problems: IPY00036337, IPY00036886.
- Added the following known problem: IPY00036815.

In the Documentation Updates section, added documentation updates to the following documents because of new features in the Service Update: *Dialogic® System Software Diagnostics Guide*, *Dialogic® IP Media Library API Library Reference*.

Document Rev 12 - published January 15, 2007

Updated for Service Update 123.

In the Release Issues section, added the following resolved problems: IPY00034254, IPY00034606.

In the Documentation Updates section, deleted the corrections for the *Dialogic® System Software Diagnostics Guide* because these corrections have been incorporated into the updated document that is now on the documentation bookshelf.

Document Rev 11 - published January 2, 2007

Updated for Service Update 122.

In the Post-Release Developments section, added Modify an Existing SIP Call Using re-INVITE for Dialogic® IPT Boards.

In the Release Issues section, added the following resolved problems: IPY00033563, IPY00033912, IPY00034036, IPY00035350, IPY00035613, IPY00035806, IPY00035822, IPY00035831.

In the Documentation Updates section, added documentation update to the *Dialogic® Global Call IP Technology Guide* that SIP call using re-INVITE is now applicable to Dialogic® IPT Boards.

Document Rev 10 - published December 18, 2006

Updated for Service Update 121.

In the Release Issues section, added the following resolved problems: IPY00033640, IPY00036025.

Document Rev 09 - published November 27, 2006

Updated for Service Update 120.

In the Post-Release Developments section:

- Added PDK Log File Detection.
- Added Media Channel Reset Capability (Stuck IP Media Channel Recovery).
- Added Global Call API Access to New H.323/Q.931 Message IEs.

In the Release Issues section, added the following resolved problems: IPY00033102, IPY00033163, IPY00033164, IPY00033472, IPY00033763, IPY00034618, IPY00034765, IPY00035148, IPY00035451.

In the Documentation Updates section:

- Added documentation update to the *Dialogic® IP Media Library API Library Reference* about new API.
- Added documentation update to the *Dialogic® Global Call IP Technology Guide* about new parameter IDs for existing parameter, IPSET_CALLINFO.

Document Rev 08 - published September 28, 2006

Updated for Service Update 116.

In the Post-Release Developments section, added On-Demand Full Reset of Dialogic® DM3 Boards.

Document Rev 07 - published September 15, 2006

Updated for Service Update 115.

In the Post-Release Developments section:

- Added New Diagnostics Management Console.
- Added New Runtime Trace Facility (RTF) Manager.

In the Documentation Updates section, added documentation update to the *Dialogic® System Release 6.1 CompactPCI for Windows® Release Guide* about requirements for diagnostic tools.

Document Rev 06 - published August 28, 2006

Updated for Service Update 113.

In the Post-Release Developments section:

- Added Support for Reporting Billing Type.
- Added Runtime Control of Double Answer for R2MF.

Document Rev 05 - published August 17, 2006

Updated for Service Update 111.

In the Post-Release Developments section, added support for tone generation and FSK for Media Channel Reset Capability (Stuck Channel Recovery).

In the Release Issues section, added the following resolved problems: IPY00033009, IPY00033059, IPY00033393, IPY00033499.

Document Rev 04 - published July 17, 2006

Updated for Service Update 108.

In the Post-Release Developments section, added Additional Supported Operating System Security Updates.

In the Release Issues section, added the following resolved problem: IPY00033058.

Document Rev 03 - published June 29, 2006

Updated for Service Update 106.

In the Release Issues section, added the following resolved problems: IPY00032897, IPY00032900.

Document Rev 02 - published June 13, 2006

Updated for Service Update 103.

In the Post-Release Developments section:

- Added Service Update for Dialogic® System Release 6.1 CompactPCI for Windows®.
- Added Media Channel Reset Capability (Stuck Channel Recovery).
- Added Notification of Layer 1 Alarm Events on SS7 Boards.
- Added Global Call Support for Time Slots on Dialogic® SS7 Boards Running in DTI Mode.

In the Release Issues section, added the following resolved problem: IPY00032793.

In the Documentation Updates section, added documentation updates for the *Dialogic® System Software Diagnostics Guide*.

Document Rev 01 - published May 26, 2006

Initial version of document.

Post-Release Developments

This section describes significant changes to the system release subsequent to the general availability release date.

1.1 Service Update for Dialogic® System Release 6.1 CompactPCI for Windows®

A Service Update for Dialogic® System Release 6.1 CompactPCI for Windows® is now available. Service Updates provide fixes to known problems, and may also introduce new functionality. New versions of the Service Update are planned to be released periodically. It is intended that this Release Update will document the features in the Service Updates.

Depending on whether you already have a version of Dialogic® System Release 6.1 CompactPCI for Windows® on your system, installing the Service Update will give you either a **full install** or an **update install**:

- If you don't have an existing version of Dialogic® System Release 6.1 CompactPCI for Windows® on your system, installing the Service Update gives you a **full install** of the release. You can select the features that you want to install, for example, DMV/DMN/DMT, Global Call Protocols, Documentation, etc.
- If you have an existing version of Dialogic® System Release 6.1 CompactPCI for Windows® on your system, installing the Service Update gives you an **update install**. The update install gives you the latest software for the features that you selected when you did the full install of the system release that is currently on your system.

1.2 PDK Support for Automatic Answer and Reject of Inbound Calls

With Service Update 201, the Protocol Development Kit (PDK) is extended to support automatic answer and reject of inbound calls.

1.2.1 Feature Implementation

With this enhancement to the PDK upon enablement through Global Call, when the **gc_DropCall( )** function is called before **gc_AnswerCall( )**, the protocol will answer and then immediately hang-up the call. This behavior enacts a full disconnect, complete with the sending of a proper signal to the switch to abandon the call.

To accomplish this, the CDP_Forced_Release_Enabled parameter is added to the respective Country Dependent Parameter (.cdp) files. This parameter controls the behavior of the protocol when **gc_DropCall( )** is called before a call is connected and allows to enable and disable this new functionality.

1.2.2 Protocol Variants

The functionality of the CDP_Forced_Release_Enabled parameter is added to the following protocols:

Protocol	Variant File
MELCAS Lineside Bidirectional	pdk_sw_e1_mcls_io.cdp
Nortel Meridian Lineside E1 Bidirectional	pdk_sw_e1_ntmd_io
United States T1 FXS/LS Bidirectional	pdk_us_ls_fxs_io
E1 CAS Bidirectional	pdk_us_mf_io
India R2 Bidirectional	pdk_in_r2_io
Argentina R2 Bidirectional	pdk_ar_r2_io
Australia R2 Bidirectional	pdk_au_r2_io.cdp
Brazil R2 Bidirectional	pdk_br_r2_io.cdp

CDP_Forced_Release_Enabled

Enable the protocol to support "forced release" of incoming calls from the offered or accepted state. The support for forcing release of incoming calls is supported under this implementation for flexibility with Global Call applications which are permitted to call gc_DropCall( ) from the Offered or Accepted state. In these states, the call will be answered transparently without notification to the application and then immediately disconnected, i.e., a "forced release" of the line. Note that in doing this, additional implications external to the PDK and Global Call might exist and should be considered, for instance call billing.

Values:

- 0 = Does not support forced release. No implicit answer will be performed transparently in this scenario, and only a PDK hang-up signal will be generated. (Default)
- 1 = Supports forced release. Calls are answered and then dropped immediately.

Refer to the *Dialogic® Global Call Country Dependent Parameters (CDP) for PDK Protocols Configuration Guide* for more information.

1.3 Status Monitor Tool Support for 16 Span Boards

Service Update 199 enhances the Status Monitor tool to support the Dialogic® DMN160TEC and DMT160TEC boards.

1.3.1 Status Monitor GUI Update

To support the 16-line Dialogic® DMN160TEC and DMT160TEC boards, the display pane of the Call Status Monitor window now contains three additional tabbed pages, each page representing the status of four lines on the board. This change retains the existing single screen StatusMon GUI. The functionality of status information, as well as the automatic status update, remains the same.

In addition to the above change, the Lineadmin button is now provided once for each board. Previously, the Lineadmin button was displayed once per line even though it provided the same information for each line.

Note: For more information about the Status Monitor tool, refer to the *Dialogic® System Software Diagnostics Guide*.

1.4 Improvement to Call Progress Analysis

Service Update 198 implements an improvement to Call Progress Analysis on Dialogic® DM3 DMV cPCI boards. Refer to the *Dialogic® System Release 6.1 CompactPCI for Windows® Release Guide* for board details.

1.4.1 Implementation

With this improvement, using Global Call and the **gc_Makecall( )** function to initiate a PSTN call with CAS or R2MF protocols that fails to complete due to call analysis detecting either a destination busy condition (destination busy tone detected), or a network congestion (fast busy/reorder tone detected) will now provide the correct call outcome. To take advantage of the feature, the **gc_ResultInfo( )** function, which provides more information about the GCEV_DISCONNECT event, will return either a GCRV_BUSY or a GCRV_CONGESTION cause value respectively. Previously, Global Call did not distinguish between a destination busy tone and a fast busy/reorder tone when performing call progress analysis under Global Call and would report destination busy for either tone.

Note: This improvement applies only to CAS and R2MF protocols.

For more information, refer to the *Dialogic® Voice API Programming Guide*, the *Dialogic® Global Call API Library Reference*, and the *Dialogic® Global Call API Programming Guide*.

1.5 Media Load Support for the Dialogic® DM/V1200A-4E1-cPCI Board

Service Update 196 adds Media Load 9B (ML9B) support for the Dialogic® DM/V1200A-4E1-cPCI board. This media load provides rich conferencing (conferencing plus echo cancellation and Tone clamping). It also disables the network front ends.

1.5.1 Feature Description

Predefined sets of features for Dialogic® DM3 Boards are provided in media loads. A media load consists of a configuration file set (PCD, FCD, and CONFIG files) and the associated firmware that is downloaded to the board. See the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* for more information about media loads.

The features and channel densities provided by Media Load 9B on the DM/V1200A-4E1-cPCI Board are as follows:

Features Supported	Network Interface (note 2)	Rich Conferencing with Echo Cancellation and Tone Clamping	Maximum Conference Size without Bridging (note 1)
Channel Density	0	120	60

Notes:1. Conference size is limited to 60 parties without bridging. Conference bridging can be used to effectively expand a conference beyond the maximum size; however, this consumes conferencing resources and reduces overall board conference density.

2. Media Load 9B disables the board's network interfaces, thus the board cannot be connected to the PSTN network. The board can be used effectively as a conferencing resource board in conjunction with network interface board(s) when properly cabled through the CT Bus.

1.5.2 Configuring the Software

The new media loads can be selected by using the Dialogic® Configuration Manager (DCM). This procedure, which must be performed before the board is started, is described in detail in the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide.*

The name of the configuration file set for Media Load 9B on the DM/V1200A-4E1-cPCI Board is **ml9b_qs2_e1**, that is, *ml9b_qs2_e1.pcd*, *ml9b_qs2_e1.fcd,* and *ml9b_qs2_e1.config*.

1.6 Handling non-2xx Responses to T.38 Switch

Service Update 194 introduces Global Call support for RFC3261 compliance for non-2xx responses to re-INVITE requests to switch to or from audio to T.38 fax and back. This feature has been tested on Dialogic® IPT Boards.

1.6.1 Feature Description

Currently, when a Global Call SIP application initiates a media type switch from/to audio or to/from fax within a dialog with a re-INVITE request, the existing media session and dialog are terminated on a non-2xx response from the UAS. RFC3261 clearly requires that the UAC keep the exiting session alive in a dialog as though the re-INVITE never occurred.

With this feature, the existing media session remains active within the dialog upon a switching request from one media type to another (fax to audio or audio to fax).

This feature is enabled by default when the application calls the **gc_ReqModifyCall( )** function or the **gc_Extension( )** function with the codec switch value. On failure to switch, the application will receive the failure events, GCEV_REQ_MODIFY_REJ/GCEV_REQ_MODIFY_FAIL and the GCEV_EXTENSION event with parm ID set to IPPARM_REJECT for set ID IPSET_SWITCH_CODEC respectively. The existing media session will be reestablished underneath and the requested local media information will be overwritten with the stored (existing) media information.

Because this feature is limited to “Manual” operating mode, an application must be configured in “Manual” mode to control the association and disassociation of media and T.38 fax devices during each call. The mode of operation is set on a board device basis. The operating mode for set ID/parm ID pair IPSET_CONFIG/IPPARM_OPERATING_MODE must be set to either of the following:

- IP_T38_MANUAL_MODE
- IP_T38_MANUAL_MODIFY_MODE

For additional information, refer to the documentation updates for Chapter 3. IP Call Scenarios and Chapter 4. IP Specific Operations in the Dialogic Global Call IP Technology Guide.

1.6.2 Manual Mode Example

This example demonstrates “Manual” mode when the switch from T.38 fax to audio is unsuccessful.

```
INT32 switchFromFaxToAudio( )
{
      GC_PARM_BLK *parmblkp = NULL;

      IP_CONNECT ipConnect;
      ipConnect.version = 0x100;
```

```
      ipConnect.mediaHandle = pline->mediaH;

      gc_util_insert_parm_ref(&parmblkp, IPSET_FOIP, IPPARM_T38_DISCONNECT,
                              (sizeof(IP_CONNECT)), (void *)(&ipConnect));

      gc_SetUserInfo(GCTGT_GCLIB_CRN, pline->crn, parmblkp, GC_SINGLECALL);

      gc_util_delete_parm_blk(parmblkp);
      /* Initiate audio codec switch */
      gc_util_insert_parm_ref(&parmblkp, IPSET_SWITCH_CODEC, IPPARM_AUDIO_INITIATE, sizeof(int),
                              NULL);

      gc_Extension(GCTGT_GCLIB_CRN,pline->crn, IPEXTID_CHANGEMODE, parmblkp, NULL, EV_ASYNC);
      gc_util_delete_parm_blk(parmblkp);
}

INT32 processEvtHandler()
{
      METAEVENT metaEvent;
      GC_PARM_BLK *parmblkp = NULL;
      GC_INFO    t_info;

      switch (evtType)
      {
          case GCEV_EXTENSIONCMPLT:
          /* received extension complete event for audio initiation*/
          /* do nothing */
          break;

          case GCEV_EXTENSION:
          /* received extension event for media readiness */
          ext_evtblkp = (EXTENSIONEVTBLK *) metaEvent.extevtdatap;
          parmblkp = &ext_evtblkp->parmblk;
          while (t_gcParmDatap = gc_util_next_parm(parmblkp, t_gcParmDatap))
          {
          switch(t_gcParmDatap->set_ID)
          {
               case IPSET_SWITCH_CODEC:
               switch(t_gcParmDatap->parm_ID)
          {
               case IPPARM_REJECT:
                gc_ResultInfo(&metaEvent,&t_info);
                gc_util_insert_parm_ref(&parmblkp, IPSET_FOIP, IPPARM_T38_CONNECT,
                                        (sizeof(IP_CONNECT)), (void *)(&ipConnect));

                gc_SetUserInfo(GCTGT_GCLIB_CRN, pline->crn, parmblkp, GC_SINGLECALL);
                break;
                case IPPARM_READY:
                /* Ready to send and receive audio */
                  gc_Listen();
                break;
           }
}
```

This example demonstrates “Manual” mode when the switch from T.38 fax to audio is unsuccessful.

```
INT32 processEvtHandler( )
{
METAEVENT metaEvent;
GC_PARM_BLK *parmblkp = NULL;
IP_CONNECT ipConnect;
GC_INFO    t_info;

switch (evtType)
{
```

```
        case GCEV_CONNECTED:
        /* received Connect event */
        /* in conversation */
        ipConnect.version = 0x100;
        ipConnect.mediaHandle = pline->mediaH;
        ipConnect.faxHandle = pline->faxH;
        ipConnect.connectType = IP_FULLDUP;
        gc_util_insert_parm_ref(&parmblkp, IPSET_FOIP, IPPARM_T38_CONNECT,
                              (sizeof(IP_CONNECT)), (void *)(&ipConnect));

        gc_SetUserInfo(GCTGT_GCLIB_CRN, pline->crn, parmblkp, GC_SINGLECALL);

        gc_util_delete_parm_blk(parmblkp);
        /* Initiate T.38 codec switch */
        gc_util_insert_parm_ref(&parmblkp,IPSET_SWITCH_CODEC,IPPARM_T38_INITIATE,
                              sizeof(int), NULL);

        gc_Extension(GCTGT_GCLIB_CRN,pline->crn,IPEXTID_CHANGEMODE, parmblkp, NULL, EV_ASYNC);
        gc_util_delete_parm_blk(parmblkp);
        break;

        case GCEV_EXTENSIONCMPLT:
        /* received extension complete event for T.38 initiation*/
        /* do nothing */
        break;

        case GCEV_EXTENSION:
        /* received extension event for media readiness */
        ext_evtblkp = (EXTENSIONEVTBLK *) metaEvent.extevtdatap;
        parmblkp = &ext_evtblkp->parmblk;
        while (t_gcParmDatap = gc_util_next_parm(parmblkp, t_gcParmDatap))
        {
        switch(t_gcParmDatap->set_ID)
        {
               case IPSET_SWITCH_CODEC:
               switch(t_gcParmDatap->parm_ID);
               {
               case IPPARM_REJECT:

                  gc_ResultInfo(&metaEvent,&t_info);

                  gc_util_insert_parm_ref(&parmblkp, IPSET_FOIP,
                                          IPPARM_T38_DISCONNECT,(sizeof(IP_CONNECT)), (void
                                          *)(&ipConnect));

                  gc_SetUserInfo(GCTGT_GCLIB_CRN, pline->crn, parmblkp, GC_SINGLECALL);

                  gc_Listen();
                  /* IPT to IPM*/
                  break;
             case IPPARM_READY:
             /* Ready to send and receive fax */
             fx_sendfax();
             break;
          }
         break;
}
```

1.6.3 Manual Modify Mode Examples

This example demonstrates “Manual” modify mode when the switch from T.38 fax to audio is unsuccessful.

```
INT32 switchFromFaxToAudio()
{
      GC_PARM_BLK *parmblkp = NULL;
      IP_CONNECT ipConnect;
      ipConnect.version = 0x100;
      ipConnect.mediaHandle = pline->mediaH;

      gc_util_insert_parm_ref(&parmblkp, IPSET_FOIP, IPPARM_T38_DISCONNECT,
                              (sizeof(IP_CONNECT)), (void *)(&ipConnect));

      gc_SetUserInfo(GCTGT_GCLIB_CRN, pline->crn, parmblkp,GC_SINGLECALL);

      gc_util_delete_parm_blk(parmblkp);
      /* Initiate audio codec switch */
      if( gc_util_insert_parm_ref(&parmblkp,GCSET_CHAN_CAPABILITY, IPPARM_LOCAL_CAPABILITY,
          sizeof(IP_CAPABILITY), &ipcap) != GC_SUCCESS )
      {
       //error
      }
      gc_ReqModifyCall (GCTGT_GCLIB_CRN,pline->crn, parmblkp, EV_ASYNC);
      gc_util_delete_parm_blk(parmblkp);
}
INT32 processEvtHandler()
{
     METAEVENT metaEvent;
     GC_PARM_BLK *parmblkp = NULL;
     switch (evtType)
     {
         case GCEV_EXTENSIONCMPLT:
         /* received extension complete event for audio initiation*/
         /* do nothing */
         break;

         case GCEV_MODIFY_CALL_ACK:
        // switch complete
          gc_Listen();
        break;

        case GCEV_MODIFY_CALL_REJ:
        case GCEV_MODIFY_CALL_FAIL:
        gc_util_insert_parm_ref(&parmblkp, IPSET_FOIP, IPPARM_T38_CONNECT,
                                (sizeof(IP_CONNECT)), (void *)(&ipConnect));
        break;
      }
```

This example demonstrates “Manual” modify mode when the switch from audio to T.38 fax is unsuccessful.

```
INT32 processEvtHandler()
{
METAEVENT metaEvent;
GC_PARM_BLK *parmblkp = NULL;
IP_CONNECT ipConnect;
switch (evtType)
{
case GCEV_CONNECTED:
     /* received Connect event */
     /* in conversation */
     ipConnect.version = 0x100;
     ipConnect.mediaHandle = pline->mediaH;
     ipConnect.faxHandle = pline->faxH;
     ipConnect.connectType = IP_FULLDUP;
     gc_util_insert_parm_ref(&parmblkp, IPSET_FOIP, IPPARM_T38_CONNECT,
                             (sizeof(IP_CONNECT)), (void *)(&ipConnect));
     gc_SetUserInfo(GCTGT_GCLIB_CRN, pline->crn, parmblkp, GC_SINGLECALL);
```

```
    if( gc_util_insert_parm_ref(&parmblkp, GCSET_CHAN_CAPABILITY,
        IPPARM_LOCAL_CAPABILITY, sizeof(IP_CAPABILITY), &ipcap) != GC_SUCCESS )
    {
    //error
    }
    gc_ReqModifyCall (GCTGT_GCLIB_CRN,pline->crn, parmblkp, EV_ASYNC);
    gc_util_delete_parm_blk(parmblkp);
    break;

    case GCEV_MODIFY_CALL_ACK:
        // Switch Complete
        fx_sendfax();
        break;

    case GCEV_MODIFY_CALL_REJ:
    case GCEV_MODIFY_CALL_FAIL:

         /* received extension event for media readiness */
         gc_util_insert_parm_ref(&parmblkp, IPSET_FOIP,
                   IPPARM_T38_DISCONNECT,(sizeof(IP_CONNECT)), (void *)(&ipConnect));

         gc_SetUserInfo(GCTGT_GCLIB_CRN, pline->crn, parmblkp, GC_SINGLECALL);

         gc_Listen();
         /* IPT to IPM*/
               break;
}
```

1.7 Important Notice about System Release Update Installation

Due to changes in the Dialogic® Software install process, an update install should not be used when updating to a more recent build from a Dialogic® System Release 6.1 CompactPCI for Windows build prior to Service Update 176. Update install does work when upgrading from Service Update 176 to any more recent build.

1.8 Media LAN Disconnection Alarm Notification for Dialogic® DM/IP Boards

With the Service Update, Dialogic® DM/IP Boards have the ability to monitor the network interface connector (NIC) and generate an alarm event if a disconnection or network failure occurs (i.e., a media LAN disconnection alarm). The event is then reported to the application via Quality of Service (QoS) alarms with the Dialogic® IP Media Library (IPML) API and the Dialogic® Global Call Alarm Management System (GCAMS), using Dialogic® Standard Runtime Library (SRL) asynchronous event notification.

Formerly, this functionality was available only on Dialogic® IPT Boards but is now supported on the following DM/IP Boards as well:

- Dialogic® DM/IP601-CPCI-100BT IP Boards
- Dialogic® DM/IP601-2E1-CPCI-100BT IP Boards

Note: The EVT_NETWORKFAILURE1 event type, which is supported on IPT Boards, is not supported on DM/IP Boards since they have only one network interface. (For information about EVT_NETWORKFAILURE1, see Section 1.25, "Modified Alarm Events for Media LAN Disconnect", on page 74 of this Release Update.)

1.8.1 Feature Description

Note: This feature is already documented in the *Dialogic® IP Media Library API Programming Guide* and *Dialogic® Global Call IP Technology Guide*; in particular, refer to:

- *Dialogic® IP Media Library API Programming Guide*: "Network Failure Alarm" section in the Quality of Service (QoS) Alarms chapter
- *Dialogic® Global Call IP Technology Guide*: "Media LAN Disconnection Alarm" section in the IP-Specific Operations chapter

For convenience, information from these sections is repeated here.

A board-level alarm notifies the application when the board's connection to the LAN has been disrupted, for example if the Ethernet cable has been disconnected or if there has been some failure in a hub or switch. When the alarm is enabled, the board checks the status of the network connection at 1-second intervals. If the board finds that the connection is disrupted, it generates a single network failure event to notify the IPML application. When a subsequent network status check indicates that the network connection has been restored, a single network failure alarm-off IPML event is generated. Both the alarm-on (failure) and alarm-off (restoration) events may also be reported to the Global Call library via GCAMS.

Note the following differences between the network failure alarm and other QoS alarms:

- The network failure alarm is a board-level alarm while most other QoS alarms operate at the channel device level.
- There are no threshold parameters associated with the network failure alarm as there are with other QoS alarms.
- The network failure alarm cannot be reset via **ipm_ResetQoSAlarmStatus( )**.
- The status of the network failure alarm cannot be queried via **ipm_GetQoSAlarmStatus( )**, and the status is not reported via **ipm_GetSessionInfo( )**.
- The network failure alarm is only reported via asynchronous notification events.

IP Media Library Considerations

The application registers for notification of the network failure alarm in much the same way as a QoS alarm, by calling **ipm_EnableEvents( )**, and deregisters via **ipm_DisableEvents( )**, and so a separate function call must be used to enable or disable the network failure alarm event using the board's device handle and EVT_NETWORKFAILURE event type. The function call will fail if a channel device handle is specified.

Event handling in IPML for the network failure alarm is identical to that for QoS alarm events, except that the handler needs to distinguish between alarm-on and alarm-off events. The event that is generated when a network failure is detected is of type IPMEV_QOS_ALARM, and it contains associated data of type IPM_QOS_ALARM_DATA. The eQoSType field of this data structure is QOSTYPE_NETWORKFAILURE, and the eAlarmState may be either ALARM_STATE_ON or ALARM_STATE_OFF.

Global Call Library Considerations

To enable a Global Call application to receive media network failure alarm events, the application must perform the following general steps:

- Explicitly open and obtain the Global Call line device handle for the IPM board device with **gc_OpenEx( )**.

  Note: It is not necessary to enable the EVT_NETWORKFAILURE event for the board device (using **ipm_EnableEvents( )**). IPML alarm event setting is taken care of automatically by the software, so the application should not enable it explicitly.

- Register the device handle (from the open operation) with GCAMS using **gc_SetAlarmNotifyAll( )**. This registration uses the wildcard alarm source object (ASO) ID, ALARM_SOURCE_ID_NETWORK_ID, because the IP Call Control library ASO ID is not known at this point.

When a media network failure alarm event occurs, the IPML library generates an IPMEV_QOS_ALARM event, which contains data that identifies the alarm as type QOSTYPE_NETWORKFAILURE. This event is processed by GCAMS, which generates a GCEV_ALARM event. When this event is received, the alarm number QOSTYPE_NETWORKFAILURE, the alarm name (the string “Network Failure”), the alarm state, the ASO ID, and the ASO name can be retrieved using standard Global Call alarm APIs.

Notes: ***1.*** The IPML ASO will pass the event to GCAMS and consume it, not reporting it directly via IPML library event.

2. By the same token, Global Call will intercept the IPML EVT_NETWORKFAILURE event, consume it and report it as a GCAMS alarm event.

Media LAN condition **prior to** application startup or alarm condition enablement is reported only in the case of a network disruption (alarm-on); that is, this notification is only for a disrupted network and not for a healthy network. Subsequent alarm events will occur as they normally would upon a change in the alarm state.

1.8.2 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about the Dialogic® IP Media Library API, see the following documents:

- *Dialogic® IP Media Library API Programming Guide*
- *Dialogic® IP Media Library API Library Reference*

For more information about the Dialogic® Global Call API in general, see the following documents:

- *Dialogic® Global Call API Programming Guide*
- *Dialogic® Global Call API Library Reference*

For features specific to IP technology, see:

- *Dialogic® Global Call IP Technology Guide*

1.9 Support for SFTP in Dialogic® Global Call SS7 Call Control Library

With the Service Update, a parameter has been added to the *gcss7.cfg* file to specify the type of file transfer protocol used by the Dialogic® Global Call SS7 server to retrieve configuration files from the Signal Interface Units (SIUs) when boards are downloaded. By default, the Global Call SS7 server uses regular ftp. This new parameter, **SIU.FTP_Type**, allows ssh ftp (sftp) to be used. For further information about the **SIU.FTP_Type** parameter, see the *Dialogic® Global Call SS7 Technology Guide*.

1.10 New Media Loads for Dialogic® DM/V4800BC Media Boards Using Special Coders

The Service Update provides new media loads for the Dialogic® DM/V4800BC Media Board. Media loads 2E and 5E provide additional, special coder support for playing files concurrently with continuous speech processing (CSP) and are only intended to be used with these new special coders. Media load 2 supports play and record, without concurrent CSP, and is also intended to be used with the special coders. Media load 1E is a basic voice and fax media load that supports a subset of the special coders.

1.10.1 Feature Description

Predefined sets of features for Dialogic® DM3 Boards are provided in media loads. A media load consists of a configuration file set (PCD, FCD, and CONFIG files) and the associated firmware that is downloaded to the board. See the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* for more information about media loads.

The new media loads are described below:

- Media Load 2E

- Media Load 5E
- Media Load 2
- Media Load 1E

Media Load 2E

Media load 2E is a voice/CSP media load for the DM/V4800BC Board. The features and channel densities provided by media load 2E are as follows:

Features Supported	Basic and Enhanced Voice; Transaction Record; ***Special Coders***	CSP with Enhanced Echo Cancellation	Special Coders (see below); Concurrent Play with CSP	CSP Streaming to CT Bus	FSK	TrueSpeech
Channel Density	208	208	208	0	0	0

Dialogic® CSP software supports the following encoding algorithms, sampling rates, and sampling sizes for playing files during a CSP streaming session in DM3 (WAVE or VOX file format):

- G.711 mu-law PCM, 8 kHz sampling rate, 8-bit resolution (64 Kbps)
- G.711 A-law PCM, 8 kHz sampling rate, 8-bit resolution (64 Kbps)

With this feature, CSP adds the following special encoding algorithms, sampling rates, and sampling sizes in media load 2E for playing files during a CSP streaming session (VOX file format only):

- G.721 at 8 kHz with 4-bit resolution (32 Kbps), with 16-bit reversal and nibble swap
- G.711 A-law and mu-law PCM, 8 kHz sampling rate, 8-bit resolution (64 Kbps) with 8-bit reversal
- G.711 A-law and mu-law PCM, 8 kHz sampling rate, 8-bit resolution (64 Kbps) with 16-bit reversal

Due to the nature of these special coders, channel density is limited.

The DX_XPB data structure settings for the new coders are:

wFileFormat	FILE_FORMAT_VOX
wDataFormat	DATA_FORMAT_G711_ALAW_8BIT_REV, or DATA_FORMAT_G711_ALAW_16BIT_REV, or DATA_FORMAT_G711_MULAW_8BIT_REV, or DATA_FORMAT_G711_MULAW_16BIT_REV, or DATA_FORMAT_G721_16BIT_REV_NIBBLE_SWAP
nSamplesPerSec	DRT_8KHZ
wBitsPerSample	8 or 4

Note: This feature does not provide additional CSP streaming encoding algorithms, sampling rates, or sampling sizes; they remain being:

- G.711 mu-law PCM, 8 kHz sampling rate, 8-bit resolution (64 Kbps)
- G.711 A-law PCM, 8 kHz sampling rate, 8-bit resolution (64 Kbps)
- Linear PCM, 8 kHz sampling rate, 16-bit resolution little Endian and big Endian format (128 Kbps)

regardless of the coders being used for playing files during the CSP streaming session.

Media Load 5E

Media load 5E is a voice/fax/CSP media load for the DM/V4800BC Board. The features and channel densities provided by media load 5E are as follows:

Features Supported	Basic and Enhanced Voice; Transaction Record; ***Special Coders***	CSP with Enhanced Echo Cancellation	Special Coders; Concurrent Play with CSP	V.17 Fax	CSP Streaming to CT Bus	FSK	TrueSpeech
Channel Density	197	197	197	15	0	0	0

Media load 5E provides the same features as media load 2E (with different densities) and also supports concurrent V.17 fax. The special coders are the same as in media load 2E. See the Media Load 2E section for more detailed information.

Media Load 2

Media load 2 is also a voice/CSP media load for the DM/V4800BC Board. The features and channel densities provided by media load 2 are as follows:

Features Supported	Basic and Enhanced Voice; Transaction Record; ***Special Coders***	CSP with Enhanced Echo Cancellation	CSP Streaming to CT Bus	FSK	TrueSpeech
Channel Density	240	240	0	0	0

Media load 2 is very similar to the existing media load 2C, however it eliminates the CSP streaming to CT Bus capability; it also adds support for the new encoding algorithms for standard playback and record as in media load 2E. See the Media Load 2E section for more detailed information.

Note: Differently from media load 2E, media load 2 does not support these special coders for playing files during a CSP streaming session; the new coders are limited to standard playbacks outside of CSP and to standard record features.

Media Load 1E

Media load 1E is a basic voice and fax media load for the DM/V4800BC Board. The features and channel densities provided by media load 1E are as follows:

Features Supported	Basic Voice; ***Special Coders***	FSK	V.17 Fax	Transaction Record
Channel Density	385	385	15	0

Note: Transaction record is not supported.

Media load 1E supports the same coders for playback and record as media load 1, and also supports the following special coders concurrent with V.17 fax:

- G.711 A-law and mu-law PCM, 8 kHz sampling rate, 8-bit resolution (64 Kbps) with 8-bit reversal
- G.711 A-law and mu-law PCM, 8 kHz sampling rate, 8-bit resolution (64 Kbps) with 16-bit reversal

The DX_XPB data structure settings for the new coders are:

wFileFormat	FILE_FORMAT_VOX
wDataFormat	DATA_FORMAT_G711_ALAW_8BIT_REV, or DATA_FORMAT_G711_ALAW_16BIT_REV, or DATA_FORMAT_G711_MULAW_8BIT_REV, or DATA_FORMAT_G711_MULAW_16BIT_REV, or
nSamplesPerSec	DRT_8KHZ
wBitsPerSample	8

1.10.2 Configuring the Software

The new media loads can be selected by using the Dialogic® Configuration Manager (DCM). This procedure, which must be performed before the board is started, is described in detail in the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide.*

The name of the configuration file set for media load 2E is **ml2e_cpciresb**, that is, *ml2e_cpciresb.pcd*, *ml2e_cpciresb.fcd,* and *ml2e_cpciresb.config*.

The name of the configuration file set for media load 5E is **ml5e_cpciresb**, that is, *ml5e_cpciresb.pcd*, *ml5e_cpciresb.fcd,* and *ml5e_cpciresb.config*.

The name of the configuration file set for media load 2 is **ml2_cpciresb**, that is, *ml2_cpciresb.pcd*, *ml2_cpciresb.fcd,* and *ml2_cpciresb.config*.

The name of the configuration file set for media load 1E reflects the channel density and so in this particular case is **ml1e_400_cpciresb**, that is, *ml1e_400_cpciresb.pcd*, *ml1e_400_cpciresb.fcd,* and *ml1e_400_cpciresb.config*.

1.10.3 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For detailed information about configuring Dialogic® DM/V4800BC Media Boards, see the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide*.

Note: The online bookshelf has not been updated for this feature, so the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* does not currently include information about these new media loads for the DM/V4800BC Board.

1.11 File Management Enhancements for DebugAngel Tool

With the Service Update, the configuration options for the Dialogic® DebugAngel tool have been enhanced to provide more capabilities for managing multiple log files.

1.11.1 Feature Description

The DebugAngel tool provides low-level firmware tracing, to aid in the troubleshooting of firmware issues on Dialogic® DM3 Boards. The tool is described in the *Dialogic® System Software Diagnostics Guide*. This feature enhances the file management capabilities for the log files created by DebugAngel. (Content of the log files remains unchanged.)

Previously, DebugAngel had only one option for naming multiple log files. With this feature, there is an additional option to include timestamps with multiple log files.

Configuration Options

In addition to command line options for starting/stopping the DebugAngel service, DebugAngel uses configuration settings specified through Windows® registry entries. These entries are created in the registry with their default settings when DebugAngel is first installed as a service (explained in the *Dialogic® System Software Diagnostics Guide*). Upon subsequent startups of DebugAngel, the values of these registry entries are loaded.

Using the Windows RegEdit.exe utility, you can modify the default settings, and the changes will take effect the next time the DebugAngel Windows service is started. The location of these entries in the registry can be found at:

```
\\HKEY_LOCAL_MACHINE\SOFTWARE\Dialogic\DebugAngel
```

Caution: Incorrect manipulation of the Windows registry can render your system unusable, requiring that you reinstall Windows. Only a system administrator qualified to modify the registry should change the DebugAngel configuration.

The following configuration options existed in the previous version of DebugAngel, and their default values remain the same. However, changes in the behavior of the **AutoRename** registry entry provide more options for the processing of multiple files.

DebugLevel
Enables/disables error or warning debug output. When set to 1, the debug output can be viewed using the Windows DebugView.exe utility. Default is 0.

LogFile
Specifies the full path and file name of the log file to be used by DebugAngel. Default is %INTEL_DIALOGIC_DIR%\log\DebugAngel.log, where %INTEL_DIALOGIC_DIR% is the value of the INTEL_DIALOGIC_DIR environment variable (for example, *C:\Program Files\Dialogic\log\DebugAngel.log*).

MaxFileSize
Specifies the maximum log file size in bytes. Default is 0 (unlimited size).

When a log file reaches the specified maximum size, the logging behavior depends on the **MaxFiles** and **AutoRename** settings as explained below.

MaxFiles
Specifies the maximum number of log files that should be created by DebugAngel. Default is 1 file. This setting is used in conjunction with the **MaxFileSize** registry entry. When a log file reaches the **MaxFileSize**, it is closed and a new log file is created. Multiple log files are named according to the **AutoRename** setting.

Notes:1. If **MaxFiles** is greater than 1, then **MaxFileSize** cannot be 0; it is not permitted to have multiple files of unlimited size. If you specify those settings, a warning message is generated and the value of the **MaxFiles** registry key is reset to 1.

2. **MaxFiles** cannot be set less than or equal to 0. If it is, a warning message is generated and the value of the **MaxFiles** registry key is reset to 1.

AutoRename
Controls whether an existing log file is backed up (e.g., when the computer is restarted), and specifies the naming convention to use when creating log files. Further information is given in the following sections:

- AutoRename Options for Single Log Files
- AutoRename Options for Multiple Log Files

AutoRename Options for Single Log Files

This section explains how the **AutoRename** options work when there is a single log file (**MaxFiles=1**).

Note: When **MaxFiles** is set to **1**, the value of the **AutoRename** registry key is automatically set to **1**.

AutoRename=0
The log file name specified in the **LogFile** registry entry is used. There is no backup of an existing log file. If the file exists when DebugAngel starts, it is deleted and replaced with a new file of the same name. When **MaxFileSize=0**, the file is allowed to grow without limit. When **MaxFileSize>0**, the file is allowed to grow to the specified limit.

When the limit is reached, the file is truncated and logging is resumed from the beginning of the file.

AutoRename=1

The log file name specified in the **LogFile** registry entry is used. If the file exists when DebugAngel starts, it is backed up and renamed with a .bak extension, to avoid overwriting the original. For example, *DebugAngel.log* is renamed *DebugAngel.log.bak*. When **MaxFileSize=0**, the file is allowed to grow without limit. When **MaxFileSize>0**, the file is allowed to grow to the specified limit. When the limit is reached, the file is truncated and logging is resumed from the beginning of the file.

AutoRename=2

Adds a timestamp with the current date and time to the log file name. A file name with a timestamp has the following format:

```
filename.MM_DD_at_hh_mm_ss.zzz.log
```

where:

- filename - the name specified in the **LogFile** registry entry, stripped of the ".log" extension. (The ".log" extension is appended to the modified file name.)
- MM - month (01=January, 02=February, 03=March, … 12=December)
- DD - day of the month (01-31)
- hh - hour (24-hour format, 00-23)
- mm - minute (00-59)
- ss - second (00-59)
- zzz - millisecond (000-999)

For example, if DebugAngel is started on February 17 at 3:11:27:357 p.m., with the **LogFile** registry setting of *DebugAngel.log*, the name of the log file created is:

```
DebugAngel.02_17_at_15_11_27.357.log
```

There is no backup of an existing log file. When **MaxFileSize=0**, the file is allowed to grow without limit. When **MaxFileSize>0**, the file is allowed to grow to the specified limit. When the limit is reached, the file is deleted, a new log file is created (using the same naming convention), and logging is resumed. This process is repeated until logging is stopped.

AutoRename=3

Adds "00" to the log file name, before the ".log" extension. (Although this option can be used with a single log file, it is more suitable when using multiple log files, where it appends an index number to each log file name.) For example, with the **LogFile** registry setting of *DebugAngel.log*, the name of the log file created is:

```
DebugAngel00.log
```

There is no backup of an existing log file. If the file exists when DebugAngel starts, it is deleted and replaced with a new file of the same name. When **MaxFileSize=0**, the file is allowed to grow without limit. When **MaxFileSize>0**, the file is allowed to grow to the specified limit. When the limit is reached, the file is truncated and logging is resumed from the beginning of the file.

AutoRename Options for Multiple Log Files

This section explains how the **AutoRename** options work when there are multiple log files (**MaxFiles>1**).

Note: When **MaxFiles** is set **greater than 1**, the value of the **AutoRename** registry key is automatically set to **3**. Furthermore, when **MaxFiles>1**, **AutoRename** must be set to either **2** or **3**. If **MaxFiles>1** and **AutoRename** is set to either **0** or **1**, a warning message is generated and the value of the **AutoRename** registry key is reset to 3.

AutoRename=2

Adds a timestamp with the current date and time to the log file name. A file name with a timestamp has the following format:

```
filename.MM_DD_at_hh_mm_ss.zzz.log
```

where:

- filename - the name specified in the **LogFile** registry entry, stripped of the ".log" extension. (The ".log" extension is appended to the modified file name.)
- MM - month (01=January, 02=February, 03=March, … 12=December)
- DD - day of the month (01-31)
- hh - hour (24-hour format, 00-23)
- mm - minute (00-59)
- ss - second (00-59)
- zzz - millisecond (000-999)

For example, if DebugAngel is started on February 17 at 3:11:27:357 p.m., with the **LogFile** registry setting of *DebugAngel.log*, the name of the *first* log file created is:

```
DebugAngel.02_17_at_15_11_27.357.log
```

There is no backup of an existing log file. When the file reaches its **MaxFileSize**, it is closed and a new log file is created. The new log file will have a timestamp with the current date and time in its file name. This process is repeated until there are **MaxFiles** log files. When **MaxFiles+1** log files are created, the oldest log file is deleted so that no more than **MaxFiles** log files are saved at any time. See Examples of Multiple Log Files below.

AutoRename=3

Adds a numeric index (counter) to the log file name. The file name has the following format:

```
filenamenn.log
```

where:

- filename - the name specified in the **LogFile** registry entry, stripped of the ".log" extension. (The ".log" extension is appended to the modified file name.)
- nn - a number starting with 00, then incrementing to 01, 02, etc., up to MaxFiles-1

For example, with the **LogFile** registry setting of *DebugAngel.log,* the names of the log files are *DebugAngel00.log*, *DebugAngel01.log*, *DebugAngel02,log*, etc.

There is no backup of an existing log file. When the file reaches its **MaxFileSize**, it is closed and a new log file is created. The new log file will have the next sequential number in its file name. This process is repeated until there are **MaxFiles** log files. When **MaxFiles+1** log files are created, the oldest log file is deleted so no more than

MaxFiles log files are saved at any time. The file naming is repeated starting with the number 00 again. See Examples of Multiple Log Files below.

Examples of Multiple Log Files

With the following settings (and default **LogFile** name):

- **AutoRename=2**
- **MaxFileSize=65536**
- **MaxFiles=5**

The resulting files in *C:\Program Files\Dialogic\log* are:

```
65,536   DebugAngel.09_26_at_16_29_08.031.log
65,536   DebugAngel.09_26_at_16_33_18.000.log
65,536   DebugAngel.09_26_at_16_44_09.008.log
65,536   DebugAngel.09_26_at_16_47_12.035.log
10,871   DebugAngel.09_26_at_16_56_58.041.log
```

Note: When this file is filled up, the first file is removed. No more than 5 files exist at any time. Each new file created has a timestamp.

With the following settings (and default **LogFile** name):

- **AutoRename=3**
- **MaxFileSize=1048576**
- **MaxFiles=4**

The resulting files in *C:\Program Files\Dialogic\log* are:

```
1,048,576   DebugAngel00.log
1,048,576   DebugAngel01.log
1,048,576   DebugAngel02.log
  650,355   DebugAngel03.log
```

Note: When this file is filled up, the first file (*DebugAngel00.log*) is overwritten. No more than 4 files exist at any time. The files are always named *DebugAngel00.log*, *DebugAngel01.log*, *DebugAngel02.log*, and *DebugAngel03.log*.

1.11.2 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about DebugAngel, see the *Dialogic® System Software Diagnostics Guide.*

1.12 File Management Enhancements for PDK Trace Tool

With the Service Update, new command line options are provided with the Dialogic® PDK Trace tool to set the output log file size and to create multiple log files.

1.12.1 Feature Description

The PDK Trace tool logs information about PDK protocols on Dialogic® DM3 Boards, to aid in the troubleshooting of protocol issues. The tool is described in the *Dialogic® System Software Diagnostics Guide*. This feature enhances the file management capabilities for the log files created by PDK Trace. (Content of the log files remains unchanged.)

Previously, PDK Trace created a single log file at application startup. Upon subsequent application startups, the existing log file could be overwritten if a log file of the same name (default or otherwise) was specified. The PDK Trace log file was also allowed to grow without limit, which could result in a file that was difficult to work with.

With this feature, you can set command line options to specify the maximum file size and to create multiple log files when the file reaches its maximum size. In addition, the log file name now shows the date and time the log was created.

Command Line Options

PDK Trace uses command line options to specify configuration settings. The following command line options have not changed and continue to be supported as described in the *Dialogic® System Software Diagnostics Guide*:

-b#
Specifies the logical ID of the board to trace (required).

-l[#] or **-l[#-#]**
Specifies which line(s) the channels to be traced are located on (optional). The default value is 1 (line 1).

-c[#] or **-c[#-#]**
Specifies which channel(s) on the specified lines to trace (optional). The default value is 1 (channel 1).

-e or **-E**
Enables CAS, R2MF, and tone-on/tone-off event tracing on supported boards (optional).

-i
Initializes the DM3 Tracer Component in the firmware (required only for the **first** time the utility is executed after the board is downloaded).

-v
Prints the version number of the utility.

-?, **-h**
Prints the help screen (command line options) for the utility.

New and Enhanced Command Line Options

Two new command line options, **-a** and **-m**, have been added for PDK Trace, and the **-f** option has been enhanced to allow log file management as follows:

-a#

Log file array size, specifies the maximum number of log files to maintain (optional). The default value is 1, and the maximum value is 10.

Example: `pdktrace -b0 -f[ExampleLog] -a5`

If left default (or explicitly set to 1), then PDK Trace creates a single log file that grows without bound (that is, no limit to the log file size).

If set greater than 1 (up to 10), then PDK Trace creates an initial log file at startup. When the log file reaches the maximum file size (either the default maximum log file size or the value specified via the **-m** command line option), the log file is closed and saved, and a new log file is created.

When the maximum number of log files (as specified by this setting) is reached, the oldest log file is deleted and a new log file is created to replace it.

Note: When the **-a** option is specified, any PDK Trace log files that exist prior to running this particular PDK Trace session are not deleted or modified in any way.

-m#

Specifies the maximum log file size in bytes (optional). The default depends on the setting of the **-a** option, as follows:

- For **-a1** (one log file), the default maximum file size is unlimited.
- For **-a2** through **-a10** (multiple log files), the default maximum file size is 100 megabytes.

The minimum that can be specified with the **-m** option is 100 kilobytes, and the maximum is 100 megabytes.

Example: `pdktrace -b0 -f[ExampleLog] -a5 -m500000`

-f[*filename*]

Specifies the name of a file on the host system to write the trace data to (optional). The default is *pdktrace.log*. The ".log" extension is appended to the specified file name string when creating the file.

If just a file name is specified, the log file will be created in the current directory where PDK Trace is being run from. However, if a path (either relative or absolute) is specified with the file name, then the log file will be created in the specified directory.

This option existed in the previous version of PDK Trace. However, the processing associated with this option has been modified to include date and time information, for management of multiple log files.

When the **-a** or **-m** option is used, the file name will automatically have a timestamp with the current date and time added to it. The ".log" extension will be appended to the end of the log file name. A file name with a timestamp has the following format:

`filename-MMDDYYYY-xxhyymzzs.log`

where:

- filename - the name specified with the **-f** option (or the default if **-f** is not used)
- MM - month (01=January, 02=February, 03=March, … 12=December)

- DD - day of the month (01-31)
- YYYY - year (e.g., 2008)
- xx - hour (24-hour format, 00-23)
- yy - minute (00-59)
- zz - second (00-59)

Example: If PDK Trace is started on February 17, 2008, at 3:11:27 p.m., with the **-f** command line option of **-f[ExampleLog]**, the name of the *first* log file created is:

```
ExampleLog-02172008-15h11m27s.log
```

When this file reaches the maximum size, it is closed and a new log file is created. The new log file will have a timestamp with the current date and time in its file name.

1.12.2 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about PDK Trace, see the *Dialogic® System Software Diagnostics Guide.*

1.13 Configuring SIP Stack Parameters with Global Call

With the Service Update, selected SIP stack parameters such as timers can now be configured with the Dialogic® Global Call API.

1.13.1 Feature Description

A new data structure, SIP_STACK_CFG, is used to configure SIP stack parameters. Details about the SIP_STACK_CFG data structure fields are given in Section 1.13.2, “SIP_STACK_CFG Data Structure”, on page 48.

To support SIP stack configuration, IP_VIRTBOARD has been updated with a new structure pointer (default is NULL) as follows:

```
typedef struct {
    ...
    ...
    /* The following is added for VIRTBOARD_VERSION_SIP_STACK_CFG support */
        SIP_STACK_CFG   *sip_stack_cfg;
    /* end VIRTBOARD_VERSION_SIP_STACK_CFG additions */
} IP_VIRTBOARD;
```

1.13.2 SIP_STACK_CFG Data Structure

The SIP_STACK_CFG structure definition has been added in the *gcip.h* file. The new data structure is described below.

Note: SIP stack parameters can only be configured once per virtual board (at **gc_Start( )**) and remain in effect throughout the Global Call application (per process).

SIP_STACK_CFG

```
typedef struct {
    unsigned long version;  /* version set by INIT_SIP_STACK_CFG */
    int retransmissionT1;
    int retransmissionT2;
    int retransmissionT4;
    int generalLingerTimer;
    int inviteLingerTimer;
    int provisionalTimer;
    int cancelGeneralNoResponseTimer;
    int cancelInviteNoResponseTimer;
    int generalRequestTimeoutTimer;
} SIP_STACK_CFG;
```

Description

The SIP_STACK_CFG data structure is used to configure selected SIP stack parameters such as timers.

The SIP_STACK_CFG data structure is referenced by the IP_VIRTBOARD data structure, which stores configuration and capability information about an IPT (virtual) board device that is populated when the device is started. An array of IP_VIRTBOARD structures (one per virtual board in the system) is referenced by the IPCCLIB_START_DATA structure, which is passed to the **gc_Start( )** function.

Applications should use the **INIT_SIP_STACK_CFG( )** function to initialize the structure with the correct version number and initial field values before setting the appropriate values.

Field Descriptions

The fields of the SIP_STACK_CFG data structure are:

version
: The version number of the data structure. The correct value is set by the **INIT_SIP_STACK_CFG( )** initialization function and should not be overridden.

retransmissionT1
: Determines several timers as defined in RFC 3261. For example, when an unreliable transport protocol is used, a Client Invite transaction retransmits requests at an interval that starts at T1 milliseconds and doubles after every retransmission. A Client General transaction retransmits requests at an interval that starts at T1 and doubles until it reaches T2. The default value is 1000.

retransmissionT2
: Determines the maximum retransmission interval as defined in RFC 3261. For example, when an unreliable transport protocol is used, general requests are retransmitted at an interval that starts at T1 and doubles until it reaches T2. If a provisional response is received, retransmissions continue but at an interval of T2. The parameter value cannot be less than 4000. The default value is 8000.

retransmissionT4
: Determines the amount of time the network takes to clear messages between client and server transactions as defined in RFC 3261. For example, when working with an unreliable transport

protocol, T4 determines the time that a UAS waits after receiving an ACK message and before terminating the transaction. The default value is 10000.

generalLingerTimer
: After a server sends a final response, the server cannot be sure that the client has received the response message. The server should be able to retransmit the response upon receiving retransmissions of the request for generalLingerTimer milliseconds. The default value is 32000.

inviteLingerTimer
: After sending an ACK for an INVITE final response, a client cannot be sure that the server has received the ACK message. The client should be able to retransmit the ACK upon receiving retransmissions of the final response for inviteLingerTimer milliseconds. The default value is 32000.

provisionalTimer
: The provisionalTimer is set when receiving a provisional response on an Invite transaction. The transaction will stop retransmissions of the Invite request and will wait for a final response until the provisionalTimer expires. If you set the provisionalTimer to 0, no timer is set, and the Invite transaction will wait indefinitely for the final response. The default value is 180000.

cancelGeneralNoResponseTimer
: When sending a CANCEL request on a General transaction, the User Agent waits cancelGeneralNoResponseTimer milliseconds before timeout termination if there is no response for the canceled transaction. The default value is 32000.

cancelInviteNoResponseTimer
: When sending a CANCEL request on an Invite request, the User Agent waits cancelInviteNoResponseTimer milliseconds before timeout termination if there is no response for the canceled transaction. The default value is 32000.

generalRequestTimeoutTimer
: After sending a General request, the User Agent waits for a final response generalRequestTimeoutTimer milliseconds before timeout termination (in this time the User Agent retransmits the request every T1, 2*T1, ... , T2, ... milliseconds). The default value is 32000.

1.13.3 Sample Code

The following example sets the SIP T1 timer to 64 ms.

```
#include "gclib.h"
..
..
#define BOARDS_NUM 1
..
..

/* initialize start parameters */
IPCCLIB_START_DATA cclibStartData;
memset(&cclibStartData,0,sizeof(IPCCLIB_START_DATA));
IP_VIRTBOARD virtBoards[BOARDS_NUM];
memset(virtBoards,0,sizeof(IP_VIRTBOARD)*BOARDS_NUM);

/* initialize start data */
INIT_IPCCLIB_START_DATA(&cclibStartData, BOARDS_NUM, virtBoards);

/* initialize virtual board */
INIT_IP_VIRTBOARD(&virtBoards[0]);

/* sip stack cfg support */
SIP_STACK_CFG sip_stack_cfg;
INIT_SIP_STACK_CFG(&sip_stack_cfg);
   virtBoard[bid].sip_stack_cfg = &sip_stack_cfg;

   sip_stack_cfg.retransmissionT1 = 64;
```

1.13.4 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about the Dialogic® Global Call API in general, see the following documents:

- *Dialogic® Global Call API Programming Guide*
- *Dialogic® Global Call API Library Reference*

For features specific to IP technology, see:

- *Dialogic® Global Call IP Technology Guide*

1.14 Disabling Automatic re-INVITE Message when Switching between Fax and Audio

When using SIP, a change between audio and fax modes may cause both sides of the call to send a re-INVITE message to renegotiate the media session between them. This would cause a glare condition, which disconnects the call.

With the Service Update, the ability to disable/enable the sending of the automatic SIP re-INVITE message upon media switch can now be configured with the Dialogic® Global Call API to prevent this glare condition.

1.14.1 Feature Description

Overview of Use Case

A user application can enable and disable the unsolicited GCEV_EXTENSION notification events associated with certain types of transition events, including media streaming connection state changes. The application can receive notification of changes in the status (connection and disconnection) of media streaming in the transmit and receive directions as GC_EXTENSIONEVT events. The events for this notification must be enabled by setting or adding the bitmask value EXTENSIONEVT_SIGNALING_STATUS to the GC_EXTENSIONEVT mask. Events can be enabled on a per board basis (using **gc_SetConfigData( )**) or on a per channel basis (using **gc_SetUserInfo( )**).

A user application needs to enable media streaming status EXTENSIONEVT_STREAMING_STATUS to get notification of media transmit and receive connected events before doing specific media tests just after another media test is completed. This is particularly useful in back-to-back testing, because in live applications there are other indications of media session events, e.g., fax CNG/CED tones, busy tone, phone hang-up tone, etc., which are not available in back-to-back testing.

For example, consider two user applications where one makes an IP call to the other, sends a fax (over IP), and after the fax session is completed, dials a string of DTMF digits for the other side to detect. If the DTMF digits are dialed before the fax session completely ends, the DTMF dial test will fail, since the media session has not switched from fax to audio yet. In order for the application to know when to dial the DTMF digits it has to know when the previous fax session has ended and the audio session has started. It knows this when it receives an event indicating that the audio media stream is connected.

When working with the **H.323** protocol, this functionality to detect the media switch is sufficient for user applications. However for **SIP** protocols, when a fax to audio switch occurs, both sides send a re-INVITE message to renegotiate the media session between them, which causes a glare condition that drops the call. This is not an expected situation for a user application.

A similar situation can occur when the media switches from audio to fax.

New Parameters to Disable/Enable Automatic re-INVITE Messages

In order to prevent this glare situation, new parameters are now available in Global Call to:

- prevent sending an automatic SIP re-INVITE when a switch from fax to audio media occurs, or when a switch from audio to fax media occurs
- re-enable the sending of an automatic SIP re-INVITE when a switch from fax to audio media occurs, or when a switch from audio to fax media occurs

The new parameter IDs are added for the existing IPSET_CONFIG set ID as shown in the following table.

Set ID	Parameter ID	Set	Send	Retrieve	SIP/ H.323
IPSET_ CONFIG	IPPARM_SIP_FAXTOAUDIO_AUTO_REINVITE_ DISABLE	**gc_SetConfigData( ) gc_SetUserInfo( )**	---	---	SIP only
	IPPARM_SIP_FAXTOAUDIO_AUTO_REINVITE_ ENABLE	**gc_SetConfigData( ) gc_SetUserInfo( )**	---	---	SIP only
	IPPARM_SIP_AUDIOTOFAX_AUTO_REINVITE_ DISABLE	**gc_SetConfigData( ) gc_SetUserInfo( )**	---	---	SIP only
	IPPARM_SIP_AUDIOTOFAX_AUTO_REINVITE_ ENABLE	**gc_SetConfigData( ) gc_SetUserInfo( )**	---	---	SIP only

By default, SIP re-INVITE messages upon media switch are sent automatically. The user application has to specifically disable the transmission of the re-INVITE by using the IPSET_CONFIG parameters. Typically, the automatic re-INVITE messages would be disabled on one side user application only, namely a fax server type of application that receives faxes.

The user application has to know whether to use this functionality depending on how the application is to be used. This is usually done at the start of an application. If the automatic re-INVITE messages are disabled on two applications in a back-to-back test, the switch from fax to audio will never occur, because neither side will send out a re-INVITE message to the other to renegotiate new media (audio) when a fax session ends.

The automatic re-INVITE messages can be disabled/enabled on a board, line, or call reference number (CRN) device basis. Code examples are shown below.

1.14.2 Sample Code

Disabling Transmission of Automatic re-INVITE on a Board Device

```
DisableFToAReinvite()
{
        LINEDEV linedevbp;
        long request_id = 0;
        GC_PARM_BLK *target_datap = NULL;

         if (gc_OpenEx(&linedevbp,":N_iptB1:P_IP",EV_SYNC, NULL) != GC_SUCCESS)
         {
             //print error
              return(FALSE);
         }

         gc_util_insert_parm_val (&target_datap, IPSET_CONFIG,
                                   IPPARM_SIP_FAXTOAUDIO_AUTO_REINVITE_DISABLE,
                                  sizeof (int), IP_MANUAL_MODE);
```

```
         if (gc_SetConfigData(GCTGT_CCLIB_NETIF,linedevbp, target_datap, 1000,
                              GCUPDATE_IMMEDIATE , &request_id, EV_ASYNC) != GC_SUCCESS)
        {
              //print error
              rcode=FALSE;
        }

        gc_util_delete_parm_blk(target_datap);

        target_datap = NULL;
}
```

Re-Enabling Transmission of Automatic re-INVITE on a Line Device

```
EnableFToAReinvite()
{
      GC_PARM_BLK *target_datap = NULL;

      if ((gc_util_insert_parm_val(&target_datap, IPSET_CONFIG,
                  IPPARM_SIP_FAXTOAUDIO_AUTO_REINVITE_ENABLE,
                  sizeof(int), NULL)) != GC_SUCCESS)
      {
        fprintf(stderr, "E%04d(%s): gc_SetUserInfo(line_dev = %d) failed, \n",Res[extts]-
>network.ts_ldev);
         fflush(stderr);
      }

       if ((gc_SetUserInfo(GCTGT_GCLIB_CHAN, Res[extts]->network.ts_ldev, target_datap,
            GC_ALLCALLS)) != GC_SUCCESS)
       {
        fprintf(stderr, "E%04d(%s): gc_SetUserInfo(line_dev = %d) failed, \n",Res[extts]-
>network.ts_ldev);
        fflush(stderr);
       }
       gc_util_delete_parm_blk(target_datap);
       target_datap = NULL;
}
```

1.14.3 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about the Dialogic® Global Call API in general, see the following documents:

- *Dialogic® Global Call API Programming Guide*
- *Dialogic® Global Call API Library Reference*

For features specific to IP technology, see:

- *Dialogic® Global Call IP Technology Guide*

1.15 IP Multicast Client Support

IP Multicast client, which was supported in previous system releases, is now supported in Dialogic® System Release 6.1 CompactPCI for Windows® on Dialogic® DM/IP Boards.

IP Multicast is a one-to-many protocol that provides a scalable solution that efficiently uses network resources and bandwidth. Unlike IP Unicast, which requires *X* copies of data to be transmitted from a source to each of *X* number of receivers, IP Multicast allows a source to transmit only a single copy of the data being sent to multiple receivers. Routers throughout the network intelligently forward the data only to those IP endpoints that have requested it.

To enable IP Multicast client when starting a session, set the **ipm_StartMedia( )** function **eDirection** parameter to **DATA_MULTICAST_CLIENT**.

Note: Alternatively, you can set the **ipm_SetRemoteMediaInfo( )** function **eDirection** parameter to **DATA_MULTICAST_CLIENT** to enable IP Multicast client. However, **ipm_SetRemoteMediaInfo( )** is deprecated and is included in the library for backwards compatibility only. Application developers should use the **ipm_StartMedia( )** function instead of **ipm_SetRemoteMediaInfo( )**.

Note: The **ipm_ModifyMedia( )** function does **not** support the **DATA_MULTICAST_CLIENT** or **DATA_MULTICAST_SERVER** direction mode. The session **eDirection** setting cannot be modified once it is set.

The following limitations apply when implementing IP Multicast client:

- For dual span DM/IP Boards, the maximum supported number of simultaneous IP media channels configured for Multicast client is limited to a TDM line (24 for T1 and 30 for E1).
- IP Multicast loopback (i.e., Multicast server and Multicast client channels on the same DM/IP Board and configured for the same Multicast group) is not supported.

For more information about the Dialogic® IP Media Library API, see the following documents:

- *Dialogic® IP Media Library API Programming Guide*
- *Dialogic® IP Media Library API Library Reference*

Code Example

The following code example shows how to start a Multicast client session. There is a subtle difference in the way the API is used when starting a Multicast client session vs. starting other types of IP sessions. When starting a Multicast client session, the IP address and coder settings of the Multicast group are specified as the **local** RTP port and **local** coder respectively. (When starting a Multicast server session, the IP address and coder settings of the Multicast group are specified as the remote RTP port and remote coder.)

```
#include <stdio.h>
#include <srllib.h>
#include <ipmlib.h>

int   nMulticastGroupPort     = 2500;
char *szMulticastGroupAddress = "225.0.0.1";
char *szDeviceName            = "ipmB1C1";

void StartMulticastClient(void)
{

   int nDeviceHandle;
   IPM_MEDIA_INFO MediaInfo;

   // Open an IP Media Channel
   nDeviceHandle = ipm_Open(szDeviceName, NULL, EV_SYNC);
   if (nDeviceHandle == -1) {
      printf("Failure Opening IP Media Channel %s", szDeviceName);
      // Perform Error Processing
   }

   // Join the IP Media Channel to a Multicast Group.  Note that the
   // Multicast Group address is specified as the Local RTP Port Information
   MediaInfo.unCount = 2;
   MediaInfo.MediaData[0].eMediaType = MEDIATYPE_AUDIO_LOCAL_RTP_INFO;
   MediaInfo.MediaData[0].mediaInfo.PortInfo.unPortId = nMulticastGroupPort;
   strcpy(MediaInfo.MediaData[0].mediaInfo.PortInfo.cIPAddress,
          szMulticastGroupAddress);

   // NOTE: For Multicast Client processing, we do not need to specify
   //       RTCP port information

   MediaInfo.MediaData[1].eMediaType = MEDIATYPE_AUDIO_LOCAL_CODER_INFO;
   MediaInfo.MediaData[1].mediaInfo.CoderInfo.eCoderType =
                                                    CODER_TYPE_G711ULAW64K;
   MediaInfo.MediaData[1].mediaInfo.CoderInfo.eFrameSize =
                                                  (eIPM_CODER_FRAMESIZE)30;
   MediaInfo.MediaData[1].mediaInfo.CoderInfo.unFramesPerPkt = 1;
   MediaInfo.MediaData[1].mediaInfo.CoderInfo.eVadEnable = CODER_VAD_DISABLE;
   MediaInfo.MediaData[1].mediaInfo.CoderInfo.unCoderPayloadType = 0;
   MediaInfo.MediaData[1].mediaInfo.CoderInfo.unRedPayloadType   = 0;

   // In Multicast Client mode, we are only going to receive data and will
   // not transmit any data.  Therefore, we don't need to specify any Remote
   // RTP/RTCP or Coder settings.

   // Start the Multicast Client Session
   if (ipm_StartMedia(nDeviceHandle,
                      &MediaInfo,
                      DATA_MULTICAST_CLIENT,
                      EV_SYNC) == -1) {
      printf("ipm_StartMedia() failed for device=\"%s\" with error=%d\n",
             ATDV_NAMEP(nDeviceHandle), ATDV_LASTERR(nDeviceHandle));
      // Perform Error Processing
   }

   /*
    .
    .
    . Continue Processing
    .
    .
    */
}
```

1.16 Troubleshooting Information for RTF Logs

To assist in troubleshooting, a table showing runtime and firmware errors that may appear in Dialogic® Runtime Trace Facility (RTF) logs is now available. You can get a description of errors and the suggested action to resolve the error. To access the table, use this link:

- *Error Code Table*

For runtime errors, the table provides the following information:

Internal error value
: The error code detected internally by the library. In some of the libraries, more than one internal error is mapped to an end user error. When contacting support about failures, this information will be helpful to the support engineer because it provides more specific information about why the error was generated. This number may appear in the RTF log (with the end user error value).

  Note: Sometimes the internal error value and end user error value are listed in the same trace entry. Sometimes the internal error value may appear as a separate entry.

End user error
: The name of the constant that is documented in the library API reference.

End user error value
: The numeric value of the constant that is documented in the library API reference. This is the value that will appear in the RTF log, which you can then search for in the table.

Description of the error
: A textual description of the error.

Action to be taken
: The suggested action to resolve the error.

For firmware errors, the table provides the following information:

Resource
: The firmware entity in which the error occurred. A resource is technically called a DM3 resource and is a software entity that provides a service to other DM3 resources. You can use the resource information to better narrow down what activity was occurring when the error occurred.

Loc hex
: The value that will appear in the RTF log (for example, 0x80000C), which you can then search for in the table.

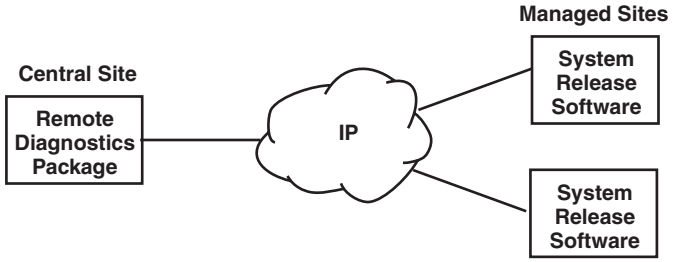
Error class
: A classification of the firmware error in broad categories. You can use this column to understand the type of action to take for a particular type of error. For example, if an error is classified as a memory error, action can be taken that is specific to this type of error (such as a pool configuration change).

Error subclass
: Provides a bit more specialization with regard to the error class. Whenever possible, if a class could be subdivided into more specific classifications, it was done. The use of the error subclass is the same as that of the error class.

Action to be taken
: The suggested action to resolve the error.

1.17 Remote Diagnostics Package

A remote diagnostics package is now available that allows you to run Dialogic® diagnostics utilities remotely from a central site. The managed sites must have Dialogic® System Release 6.1 CompactPCI for Windows® installed. The central site does **not** need Dialogic® System Release 6.1 CompactPCI for Windows® installed.



The remote diagnostics package is a *subset* of the system release software. It is designed for managing multiple remote sites from a central site, where the central site does not need the system software release or any Dialogic® boards installed. Instead, the remote diagnostics package must be installed at the central site. The diagnostics utilities in the remote diagnostics package are the same as the diagnostics utilities in Dialogic® System Release 6.1 CompactPCI for Windows®.

1.17.1 Diagnostics Utilities

The remote diagnostics package includes the following utilities:

- Diagnostics Management Console (DMC)
- Runtime Trace Facility Manager and Server application (RTFManager, RTFServer)

For information about these utilities, see the *Dialogic® System Software Diagnostics Guide* on the online bookshelf for Dialogic® System Release 6.1 CompactPCI for Windows®.

1.17.2 Installing the Remote Diagnostics Package

The remote diagnostics package can be downloaded from the Dialogic support website.

Requirements at central site:

- SSH client
- IP connectivity to managed sites
- Java Runtime Environment (JRE) version 1.5 or later

Requirements at managed sites:

- SSH server
- IP connectivity to central site
- Dialogic® System Release 6.1 CompactPCI for Windows® installed

1.18 Enhanced Diagnostics Tools

The Service Update introduces enhanced versions of the following diagnostics tools:

- PSTN Diagnostics (pstndiag)
- Status Monitor (statusmon)

Notes:1. Java Runtime Environment (JRE) version 1.5 or later must be installed on your system to run the new diagnostics tools.

2. These tools are used with Dialogic® DM3 Boards only.

1.18.1 PSTN Diagnostics (pstndiag)

The PSTN Diagnostics tool (pstndiag) is a utility for diagnosing and troubleshooting call control issues on public switched telephone network (PSTN) connections.

The pstndiag tool has a graphical user interface (GUI). When you start the tool, a tree view of all installed Dialogic® DM3 Boards is displayed. The view can be expanded to show the lines (trunks) on each board and the channels on each line. At each level (board, line, channel), different diagnostics activities can be launched, for example:

- At the board level, you can display board configuration (board name, board number, number of lines, number of channels per line, and signaling type). You can also launch the statusmon tool. (The new statusmon tool is described in Section 1.18.2, “Status Monitor (statusmon)”, on page 60.)
- At the line level, you can launch the lineadmin tool to put lines in/out of service, generate transmit alarms, enable/disable various types of loopbacks, and report bipolar violations, consecutively errored seconds, frame errors, and other saturation alarms.
- At the channel level, you can launch the phone tool to perform call control operations. You can also trace all call related activity on a given channel and store it in a columnar format based on timestamp deltas.

Running the PSTN Diagnostics Tool

To run the **new** version of pstndiag, enter the command:

- `pstndiag -j`

(The previous version of the tool is still supported and can be run by entering the command `pstndiag` without the `-j`.)

The new version of pstndiag includes the following changes:

- Faster startup
- Changes in the board tree view
- Additional features in the lineadmin tool: enabling all supported loopback modes and counters for saturation alarms
- Configurable modes of operation for the phone tool: basic, advanced, and expert

Note: More detailed information about the new version of pstndiag will be provided in the *Dialogic® System Software Diagnostics Guide*, which is scheduled to be updated soon.

1.18.2 Status Monitor (statusmon)

The Status Monitor tool (statusmon) is a utility for monitoring the current activity on all lines and channels on a Dialogic® DM3 Board. The primary use case is as a long-term monitoring tool.

The statusmon tool displays the following information:

- Alarm status (red, yellow, LOS)
- Channel state
- Call state

Running the Status Monitor Tool

The statusmon tool is typically launched from pstndiag, but it can also be run on its own. To run the **new** version of statusmon, enter the command:

- `run_statusmon.sh -board #`

where # is the logical board number of the board to monitor.

(The previous version of the tool is still supported and can be run by entering the command `statusmon board` or `statusmon board trunk channel`.)

The new version of statusmon includes the following changes:

- No line (trunk) or channel mode. However, these capabilities are supported via the pstndiag tool.

Note: More detailed information about the new version of statusmon will be provided in the *Dialogic® System Software Diagnostics Guide*, which is scheduled to be updated soon.

1.19 New QoS Alarms for RTCP and RTP Inactivity

Dialogic® IPT Boards monitor for various Quality of Service (QoS) alarms such as excessive average jitter, percentage of lost packets, and RTP packet latency. With the Service Update, two new QoS alarms for Dialogic® IPT Boards are provided:

QOSTYPE_RTCPTIMEOUT
QoS alarm for Real Time Control Protocol (RTCP) inactivity

QOSTYPE_RTPTIMEOUT
QoS alarm for Real Time Protocol (RTP) inactivity

Note: The two new QoS alarms are supported on Dialogic® IPT Boards only.

The purpose of these alarms is to indicate when transmission of RTCP and RTP packets from the remote endpoint has stopped completely (i.e., when the IP network fails), as opposed to an intermittent interruption (which can be monitored through the lost packets alarm, for example).

If an RTCP or RTP timeout alarm is received, the application can take action to release SIP/H.323 channels that remain "hung," without having to restart the application to recover. The application is responsible for assessing the RTCP and RTP timeout situation and determining the appropriate response. The timeout values are configurable.

1.19.1 Implementing QoS Alarms

The following sections provide guidelines for implementing QoS alarms in your application. Either the Dialogic® Global Call API or Dialogic® IP Media Library API can be used for this purpose.

- Implementing QoS Alarms Using Dialogic® Global Call API
- Implementing QoS Alarms Using Dialogic® IP Media Library API

1.19.1.1 Implementing QoS Alarms Using Dialogic® Global Call API

The following steps provide guidelines for implementing QoS alarms in your application using the Dialogic® Global Call API. For details on the Dialogic® IP Media Library data structures that are mentioned, see the *Dialogic® IP Media Library API Library Reference*. For details on the Global Call functions that are mentioned, see the *Dialogic® Global Call API Library Reference*. In addition, the *Dialogic® Global Call IP Technology Guide* provides information about using Global Call to implement QoS alarms.

1. Optional steps before enabling a QoS alarm:
   a. Call **gc_GetAlarmParm( )** to retrieve the current settings of QoS parameters on the specified IP channel.
   b. If you need to change current QoS parameter values, set up the IPM_QOS_THRESHOLD_INFO structure with desired values. This structure contains one or more IPM_QOS_THRESHOLD_DATA structures. Note that you must explicitly specify the value for every parameter in the IPM_QOS_THRESHOLD_DATA structure, even if you want to use the default

value for some parameters and non-default values for other parameters. The default settings for the two new QoS alarm types are:

QoS Type	Time Interval (ms)	Debounce On (ms)	Debounce Off (ms)	Fault Threshold	%Success Threshold	%Fail Threshold
QOSTYPE_ RTCPTIMEOUT	0	0	0	250(x100 msec = 25 sec)	0	0
QOSTYPE_ RTPTIMEOUT	0	0	0	1200(x100 msec = 120 sec)	0	0

Note: Only the fault thresholds are configurable; all other parameters must be set to 0. For information about setting the fault thresholds, see Section 1.19.2, "Setting the Fault Thresholds", on page 63.

c. Call **gc_SetAlarmParm( )** to use the QoS parameter values set in step 1b.

2. Call **gc_SetAlarmNotifyAll( )** to enable notification of all QoS alarms, or use **gc_SetAlarmConfiguration( )** to enable notification of selected QoS alarms. (QoS monitoring for the new QoS alarms is disabled by default.)
3. Monitor QoS alarm notification events:
   a. When a QoS alarm has been triggered, a GCEV_ALARM event is generated by the system. Call the Standard Runtime Library function **sr_getevttype( )** to return the event type.
   b. Use the following Global Call functions to extract further details about the alarm: **gc_GetMetaEvent( )**, **gc_AlarmSourceObjectID( )**, **gc_AlarmNumber( )**, **gc_AlarmName( )**.

Note: For Dialogic® IPT Boards, the system software sends a QoS alarm event containing ALARM_STATE_ON when a fault threshold is exceeded, but does **not** report a QoS event containing ALARM_STATE_OFF when the threshold returns to the programmed level. An RTCP/RTP timeout alarm is reset either on resumption of RTCP/RTP packets, end of RTP session, or end of call, but no notification is sent to the application when the alarm is reset.

1.19.1.2 Implementing QoS Alarms Using Dialogic® IP Media Library API

The following steps provide guidelines for implementing QoS alarms in your application using the Dialogic® IP Media Library API. For details on the IP Media Library functions and data structures that are mentioned, see the *Dialogic® IP Media Library API Library Reference*. In addition, the *Dialogic® IP Media Library API Programming Guide* describes how QoS may be used in an application.

1. Optional steps before enabling a QoS alarm:
   a. Call **ipm_GetQoSThreshold( )** to retrieve the current settings of QoS parameters on the specified IP channel.
   b. If you need to change current QoS parameter values, set up the IPM_QOS_THRESHOLD_INFO structure with desired values. This structure contains one or more IPM_QOS_THRESHOLD_DATA structures. Note that you must explicitly specify the value for every parameter in the IPM_QOS_THRESHOLD_DATA structure, even if you want to use the default

value for some parameters and non-default values for other parameters. The default settings for the two new QoS alarm types are:

QoS Type	Time Interval (ms)	Debounce On (ms)	Debounce Off (ms)	Fault Threshold	%Success Threshold	%Fail Threshold
QOSTYPE_ RTCPTIMEOUT	0	0	0	250(x100 msec = 25 sec)	0	0
QOSTYPE_ RTPTIMEOUT	0	0	0	1200(x100 msec = 120 sec)	0	0

Note: Only the fault thresholds are configurable; all other parameters must be set to 0. For information about setting the fault thresholds, see Section 1.19.2, "Setting the Fault Thresholds", on page 63.

c. Call **ipm_SetQoSThreshold( )** to use the QoS parameter values set in step 1b.

2. Enable QoS alarms and start media streaming:
   a. Call **ipm_EnableEvents( )** to enable QoS monitoring for a list of alarm types. (QoS monitoring for the new QoS alarms is disabled by default.) The events for the two new QoS alarm types are:
   EVT_RTCPTIMEOUT - timeout event indicating RTCP packets are no longer being received
   EVT_RTPTIMEOUT - timeout event indicating RTP packets are no longer being received
   b. Call **ipm_StartMedia( )** to start media streaming and begin QoS monitoring.
3. Monitor QoS alarm notification events:
   a. When a QoS alarm has been triggered, an IPMEV_QOS_ALARM event is generated by the system. Call the Standard Runtime Library function **sr_getevttype( )** to return the event type.
   b. Use Standard Runtime Library API functions such as **sr_getevtdatap( )** to query the IPM_QOS_ALARM_DATA structure to learn whether the alarm state is on or off.

   Note: For Dialogic® IPT Boards, the system software sends a QoS alarm event containing ALARM_STATE_ON when a fault threshold is exceeded, but does **not** report a QoS event containing ALARM_STATE_OFF when the threshold returns to the programmed level. An RTCP/RTP timeout alarm is reset either on resumption of RTCP/RTP packets, end of RTP session, or end of call, but no notification is sent to the application when the alarm is reset.
4. Perform clean-up activities:
   a. Call **ipm_Stop( )** to stop media streaming.
   b. Call **ipm_DisableEvents( )** to stop QoS parameter monitoring.

1.19.2 Setting the Fault Thresholds

The fault thresholds in the IPM_QOS_THRESHOLD_DATA structure for the RTCP and RTP timeouts can be set as follows:

IPM_QOS_THRESHOLD_DATA Structure Parameter	Minimum Fault Threshold	Maximum Fault Threshold	Default Fault Threshold
RTCP timeout unFaultThreshold	50	1200	250
RTP timeout unFaultThreshold	50	1200	1200
Note: Fault threshold unit is 100 msec. Threshold values range from 5 seconds (50 x 100 msec) to 120 seconds (1200 x 100 msec) with a resolution of 1000 msec.			

The threshold values can be modified at any time. If the channels being modified are in a call, the new values take effect as soon as an RTCP/RTP packet is received for that session. If the call is disconnected right at that moment, the new values take effect at the start of the next call.

Guidelines for Setting the Fault Thresholds

An RTP timeout alarm is **not** generated under the following conditions (although RTCP timeout alarms may be generated):

- if Voice Activity Detection (VAD) is on
- if stream mode is set to send-only or inactive (due to SIP re-INVITE)
- if RTCP timeout alarm monitoring is enabled but an RTCP timeout has not occurred. The RTP timeout alarm will only be generated after an RTCP timeout alarm has occurred.

By making RTP timeouts contingent upon RTCP timeouts being disabled, false RTP timeouts are intended to be blocked. Dialogic® IPT Boards working with terminals that handle RTCP should use RTCP timeouts to detect unplanned interruptions, while Dialogic® IPT Boards working with terminals that only handle RTP should have RTCP timeouts disabled.

When alarm monitoring is enabled for both RTCP and RTP, if the RTCP timeout unFaultThreshold is greater than the RTP timeout unFaultThreshold, the actual RTP timeout will be equivalent to the RTCP timeout.

Note that when alarm monitoring is *enabled* for RTP and *disabled* for RTCP, there is a possible scenario where the RTP timeout alarm may be received **after** the threshold set by the application. This will happen only when the RTCP timeout unFaultThreshold is greater than the RTP timeout unFaultThreshold. The Dialogic® IPT Board firmware still waits for the RTCP threshold to be reached before generating an RTP alarm, even though the RTCP alarm is not enabled and will not be sent to the application. For example:

- RTP timeout unFaultThreshold is set to 100, and RTP timeout alarm monitoring is enabled.
- RTCP timeout unFaultThreshold is set to its default value (250), and RTCP timeout alarm monitoring is disabled.

In such a setup, the application would expect to be notified of RTP timeout if RTP stops for 10 seconds, but the application will receive the RTP timeout alarm after 25 seconds. To prevent this from happening and to make sure that RTP timeout alarms are received at

the expected time, set RTP timeout unFaultThreshold higher than RTCP timeout unFaultThreshold.

1.19.3 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about the Dialogic® Global Call API, see the following documents:

- *Dialogic® Global Call IP Technology Guide*
- *Dialogic® Global Call API Library Reference*

For more information about the Dialogic® IP Media API, see the following documents:

- *Dialogic® IP Media Library API Programming Guide*
- *Dialogic® IP Media Library API Library Reference*

1.20 New Operating System Support

In addition to the supported operating systems listed in the *Dialogic® System Release 6.1 CompactPCI for Windows® Release Guide*, Windows® Server 2003 SP2 is now supported with this Service Update.

1.21 Support for Compute Platforms

With the Service Update, Dialogic® System Release 6.1 CompactPCI for Windows® has been validated on the following chassis/Single Board Computers (SBC):

- ADLINK cPCIS-3320/AC with cPCI-6910 SBC
- Advantech MIC-3081B with MIC-3369C SBC
- Diversified Technologies PlexSys-4 with CPB4612 SBC
- Performance Technologies CPC5505-B3M3H1 SBC

Support for Peripheral Hot Swap (PHS) and Redundant Host (RH) using the Pigeon Point Systems Hot Swap Kit has **not** been specifically validated for these chassis. For a list of chassis/SBC that support PHS and RH, refer to the *Dialogic® System Release 6.1 CompactPCI for Windows® Release Guide*.

1.22 New Parameter for Adjusting Silence Threshold on Dialogic® DM3 Boards

With the Service Update, the user has the ability to adjust the silence threshold parameter on Dialogic® DM3 Boards to a value above or below the default value of -43 dBm0 while using play and record functions like **dx_play( )**, **dx_record( )**, and **ec_reciottdata( )**. For instance, its adjustment affects the threshold for silence termination conditions in the Dialogic® R4 API TPT structure. It also affects silence detection via R4 unsolicited SRL events.

The silence threshold is the level that defines whether the incoming data to a voice channel is recognized as silence or non-silence. The threshold is defined by the minimum energy level of a signal below which it is considered as silence. With this new feature, the user can statically adjust the silence threshold default value of -43 dBm0 via the DM3 firmware configuration file across all voice channels on a Dialogic® DM3 Board.

Configuration Example

To change the default value of the silence threshold, you must add a new parameter in the CONFIG file that was selected for your board. The parameter is **0x70B**, and must be added in the [sigDet] section of the CONFIG file. A value equal to the desired silence threshold, measured in dBm0, must be entered. For example:

```
[sigDet]

SetParm=0x70B, 0xffd3 ! SD_ParmMinEnergy in dBm0 (e.g. 0xffd3=-45, 0xffda=-38,
                        Default: 0xffd5=-43)
```

After the CONFIG file is saved, the changes take effect after downloading.

For further information about modifying DM3 CONFIG files, see the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide*.

1.23 File Management Enhancements for ISDNtrace Tool

With the Service Update, the user can specify new command line options provided with the ISDNtrace tool to set the output log file size and to create multiple backup log files to be archived.

1.23.1 Feature Description

This feature enhances the existing ISDN tracing file management for boards configured with an ISDN load. Currently, all data is logged to a single file that can get too large during a session, and the batch operations can copy over files that might be needed. With this feature, the user can set command line options for size so that the single file is a manageable size, and also can set options to create multiple log files when the file reaches the designated file size. In addition, the standard log file name format now

conveniently shows the date and time the log was created. The user also has an option to disable logging to STDOUT to help manage trace output.

New Command Line Options

Currently, the ISDNtrace tool supports the following command line options as described in the *Dialogic® System Software Diagnostics Guide*:

```
syntax: isdntrace -b# [-f xxxx] [-d#]
```

-b<n>
Logical ID of board (required). Use the listboards utility (Linux) or the Dialogic® Configuration Manager (DCM) (Windows) to obtain the board's logical ID.

Note: The listboards utility is described in the Administration Guide for the Dialogic® Software release, and the configuration manager is described in the Configuration Guides for the Dialogic® Software (Windows) release.

-d<n>
The D-channel number (trunk number) on the specified board. The default value is 1.

-f <file>
Output log file name (required to save output in a file).

Note: A space is used after the -f option but not after -b or -d options.

-h
displays the same help information available in the ISDNtrace help menu screen. Note that this option does not show on the syntax above; however it is available.

For the ISDNtrace tool, new command line options have been added and the -f option enhanced to allow the user to manage log file(s) as follows:

```
syntax: isdntrace [-a#] -b# [-d#] [-f xxxx] [-m#] [-s]
```

-a<n>
Log file array size, max=10, default=1, optional

-f <file>
Enable logging to file, optional

Note: A space is used after the -f option but not after -a, -b, -d, -m or -s options.

-m<n>
Max log file size (expressed in bytes; for example, 500,000 bytes is specified as -m500000), optional

Min=100 Kilobytes, max=100 Megabytes

Default=unlimited if log file array size=1, else 100 Megabytes

-s
Disable logging to STDOUT, optional

Details about these command line options follow:

-a<n>

This command line option allows the user to specify the maximum number of log files to maintain.

The user can specify a log file array size between 1 and 10. By default, the number of log files to be archived is 1. If the user specifies the -f command line option but does not specify this option (or specifies it with an array size of 1), then ISDNtrace creates a single log file that grows without bound (that is, no limit to the log file size).

If the user specifies this option with an array size greater than 1 (but less than or equal to 10), then ISDNtrace creates an initial log file at startup. When the log file reaches the maximum file size (either the default maximum log file size or the value specified via the -m command line option), the log file is closed and a new log file is created.

Whenever ISDNtrace attempts to open a new log file, it first checks to see if the current number of log files created is equal to the number of files specified for the log file array. If not, then the new log file is created. Otherwise, the oldest log file is deleted and a new log file is created to replace it.

It should be noted that any ISDNtrace log files that exist prior to running the ISDNtrace tool are not deleted or modified in any way. Due to the new log file naming convention (see -f option), all ISDNtrace log files have unique timestamps in their log file names and are not overwritten when ISDNtrace starts up.

-f <file>

This option existed in the previous versions of ISDNtrace. However, the processing associated with this option has been modified to include date and time information.

This command line option specifies the log file name of the log file into which the trace can be captured. If this option is not specified on the command line, then no trace output will be saved to a log file.

The naming of ISDNtrace log files has been modified to fit the following format:

```
<File>-MMDDYYYY-xxhyymzzs.log
```

where:

- MM - current month (01=Jan, 02=Feb, 03=Mar, … 12=Dec)
- DD - current day of the month
- YYYY - current year (e.g., 2006)
- xx - current hour in day (24 hour format, 00-23)
- yy - current minute in hour (00 - 59)
- zz - current second in minute (00 - 59)

In the description above, the log file name is what the user specified on the command line. If the user specifies a -f command line option as the last parameter on the command line and does not specify a log file name, then the default log file name of ISDNTRACE will be used.

Note: In order to get a default log file name, the -f option has to be used at the end of the command line.

For example, if the user started ISDNtrace specifying the -f command line option without a log file name on January 17, 2007 at 03:11:27 pm, the log file created would be:

```
isdntrace-01172007-15h11m27s.log
```

Alternatively, the user can specify the -f command line option with a log file name specified as in the following example:

```
isdntrace -b0 -f test
```

In this example, if ISDNtrace was started on January 17, 2007 at 03:11:27 pm, the resultant log file name would be:

```
test-01172007-15h11m27s.log
```

It should be noted that since the log file name created by ISDNtrace has a .log extension appended to it, if the user specifies a log file name with a .log extension already appended to it, the resultant log file name will have the date and time inserted between the root log file name and the extension. For example, if the user issued the following command line:

```
isdntrace -b0 -f 4ess_test.log
```

Then the resultant log file name would be:

```
4ess_test-01172007-15h11m27s.log
```

-m<n>

The -m command line option is used to specify the maximum log file size. By default, the maximum log file size is 100 Megabytes. The valid range that can be specified for the maximum log file size is from 100 Kilobytes up to 100 Megabytes.

The format of the file size is specified as a long integer value. For example, to specify a maximum log file size of 250,000 bytes, the following command line should be specified:

```
isdntrace -b0 -m250000 -f test.log
```

It should also be noted that the -m command line option will have no effect if the log file array size is 1, in which case the log file will be allowed to grow in size without limit.

-s

The -s command line option can be specified to prevent trace output to STDOUT. When ISDNtrace attempts to capture a large amount of trace information in a short amount of time, its processing can fall behind if trace output is displayed to STDOUT. This will result in “enqueue fail” failures and the loss of trace information as seen in the example below:

```
                                Tue Jan 16 17:30:58 2007
                                TX Frame: Time = 2428.372
                                Command=1   SAPI=0x00
                                TEI=0x00
                                0x01 0xe6  Receive Ready
                                Hex Dump:
                                02 01 01 e6
Enqueue Failed

Tue Jan 16 17:30:58 2007
RX Frame: Time = 2428.372
Command=1   SAPI=0x00
TEI=0x00
0xe6 0xce  Information
PD=0x08 Dest=0 CR=0x1e2a
CALL DISCONNECT(0x45)
  1:           CAUSE(0x08)
  2:           IE Length(0x02)
  3:  1------- Extension Bit
      -00----- Coding Standard
```

```
      ---0---- Spare
      ----0010 Location
  4:  1------- Extension Bit
      -0010000 Cause Value
Hex Dump:
02 01 e6 ce 08 02 1e 2a 45 08
02 82 90
Enqueue Failed
```

In order to avoid loss of trace information and provide more robust performance of the ISDNtrace tool, the -s command line option should be specified to disable trace output to STDOUT whenever the capture of trace information for a large amount of calls is being performed, or “enqueue fail” failures occur.

1.23.2 Supported Boards

The following boards support this feature:

- Dialogic® DM3 Boards with network interface

1.23.3 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about the ISDNtrace tool, see the *Dialogic® System Software Diagnostics Guide*.

1.24 New Media Loads for Dialogic® DM/V2400A-cPCI and DM/V4800BC Media Boards

The Service Update provides new media loads for the Dialogic® DM/V2400A-cPCI and DM/V4800BC Media Boards. These new media loads support a combination of voice with conferencing and/or fax.

Media load 9F, for the DM/V2400A-cPCI and DM/V4800BC Boards, supports rich conferencing (conferencing, echo cancellation, and tone clamping) and fax. Media load 9F-MC, for the DM/V4800BC Board only, provides a similar set of features and also allows a larger conference size (with fewer conferences). Media load 10F, for the DM/V4800BC Board only, is a high density basic voice and rich conferencing media load.

1.24.1 Feature Description

Predefined sets of features for Dialogic® DM3 Boards are provided in media loads. A media load consists of a configuration file set (PCD, FCD, and CONFIG files) and the associated firmware that is downloaded to the board. See the *Dialogic® DM3 Architecture*

for CompactPCI on Windows® Configuration Guide for more information about media loads.

The new media loads are described below (the channel densities of the features in media load 9F are different on the DM/V2400A-cPCI and DM/V4800BC Media Boards, so they are discussed separately):

- Media Load 9F for DM/V2400A-cPCI Media Board
- Media Load 9F for DM/V4800BC Media Board
- Media Load 9F-MC for DM/V4800BC Media Board
- Media Load 10F for DM/V4800BC Media Board

Media Load 9F for DM/V2400A-cPCI Media Board

The features and channel densities provided by media load 9F on the DM/V2400A-cPCI Media Board are as follows:

Features Supported	Rich Conferencing with Echo Cancellation and Tone Clamping	Maximum Conference Size without Bridging	Fax
Channel Density	120	60	15

Note: Conference size is limited to 60 parties without bridging. Conference bridging can be used to effectively expand a conference beyond the maximum size. Conference bridging consumes conferencing resources, reducing overall board conference density.

Media Load 9F for DM/V4800BC Media Board

The features and channel densities provided by media load 9F on the DM/V4800BC Media Board are as follows:

Features Supported	Rich Conferencing with Echo Cancellation and Tone Clamping	Maximum Conference Size without Bridging	Fax
Channel Density	256	16	30

Note: Conference size is limited to 16 parties without bridging. Conference bridging can be used to effectively expand a conference beyond the maximum size. Conference bridging consumes conferencing resources, reducing overall board conference density.

Media Load 9F-MC for DM/V4800BC Media Board

The features and channel densities provided by media load 9F-MC on the DM/V4800BC Media Board are as follows:

Features Supported	Rich Conferencing with Echo Cancellation and Tone Clamping	Maximum Conference Size without Bridging	Fax
Channel Density	250	Mixed: One with conference size of 90, and 10 with conference size of 16 each	30

Media load 9F-MC supports a total of 11 DSP conference resources or DCB devices, each represented in the Audio Conferencing (DCB) API as a device dcbB*n*D*y*. It is the **last** DSP/DCB device with the maximum conference size of 90. Use the **dcb_dsprescount( )** function to obtain the available conference resource count for a specified DSP.

To differentiate with the previous media load 9F, the string -MC is added to the pcd and configuration file names.

Note: For the smaller sized conferences, conference size is limited to 16 parties without bridging. Conference bridging can be used on the smaller sized conferences to effectively expand a conference beyond the maximum size. Conference bridging consumes conferencing resources, reducing overall board conference density.

Media Load 10F for DM/V4800BC Media Board

The features and channel densities provided by media load 10F on the DM/V4800BC Media Board are as follows:

Features Supported	Basic Voice	FSK	Transaction Record	Rich Conferencing with Echo Cancellation and Tone Clamping	Maximum Conference Size without Bridging
Channel Density	420	420	0	32	16

There are 420 total voice resources. Any combination of the voice features (basic voice and FSK) can be used up to a total of 420. (Transaction record is not supported.) In addition to these voice resources, 32 conferencing resources (with echo cancellation and tone clamping) can be used.

Notes:1. The ability to sustain 100% load plays/records on all 420 voice channels is dependent upon several factors such as the audio length and the simultaneity of the playbacks and records. Heavy simultaneous plays or records above ~ 300 channels can cause quality degradation on the audio in these particular cases:

- Long audio lengths; the longer the audio, the more likely degradation can exist.
- Simultaneous voice I/O operation start on all channels; if all channels are instructed to start the I/O operation at the same time, it will likely cause noticeable degradation.
- Records are more stressful on the resources and are more likely to produce degradation than playbacks.

Under normal field conditions, calls are usually spread among channels at different start times, causing a natural staggering of the I/O operations. Also, they are typically of different lengths and may terminate at different times. This statistical behavior of calls

reduces the call duty cycle, making audio degradation less likely even at full 420 channel voice density.

2. The maximum bit rate for standard play/record with this media load is 64 Kbps. The following coders, which are normally part of the standard basic voice media load (ML1), are not supported with ML10F:
   - Linear PCM, 8 KHz sampling rate, 16-bit resolution (128 Kbps) VOX and WAVE
   - Linear PCM, 11 KHz sampling rate, 8-bit resolution (88 Kbps) VOX and WAVE
   - Linear PCM, 11 KHz sampling rate, 16-bit resolution (176 Kbps) VOX and WAVE
3. Conference size is limited to 16 parties without bridging. Conference bridging can be used to effectively expand a conference beyond the maximum size. Conference bridging consumes conferencing resources, reducing overall board conference density.

1.24.2 Configuring the Software

The new media loads can be selected by using the Dialogic® Configuration Manager (DCM). This procedure, which must be performed before the board is started, is described in detail in the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide.*

The name of the configuration file set for media load 9F on the DM/V2400A-cPCI Board is **ml9f_cpcires**, that is, *ml9f_cpcires.pcd*, *ml9f_cpcires.fcd,* and *ml9f_cpcires.config*.

The name of the configuration file set for media load 9F on the DM/V4800BC Board is **ml9f_cpciresb**, that is, *ml9f_cpciresb.pcd*, *ml9f_cpciresb.fcd,* and *ml9f_cpciresb.config*.

The name of the configuration file set for media load 9F-MC on the DM/V4800BC Board is **ml9f-MC_cpciresb**, that is, *ml9f-MC_cpciresb.pcd*, *ml9f-MC_cpciresb.fcd,* and *ml9f-MC_cpciresb.config*.

The name of the configuration file set for media load 10F on the DM/V4800BC Board is **ml10f_cpciresb**, that is, *ml10f_cpciresb.pcd*, *ml10f_cpciresb.fcd,* and *ml10f_cpciresb.config*.

1.24.3 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about the Dialogic® Audio Conferencing (DCB) API, see the following documents:

- *Dialogic® Audio Conferencing API Programming Guide*
- *Dialogic® Audio Conferencing API Library Reference*

For detailed information about configuring Dialogic® DM/V2400A-cPCI and DM/V4800BC Media Boards, see the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide*.

Note: The online bookshelf has not been updated for this feature, so the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* does not currently include information about media loads 9F, 9F-MC, and 10F.

1.25 Modified Alarm Events for Media LAN Disconnect

With the Service Update, the user receives LAN disconnect alarm events for each Ethernet port on the Dialogic® IPT Boards, with indications specifically for when both ports fail and if either recovers.

1.25.1 Feature Description

The Dialogic® IPT Boards have dual 1000BT and 100BT Ethernet ports, also referred to as network interface connectors (NICs). The board supports LAN disconnect alarm monitoring and reporting, which provides that the board generate an alarm event if a disconnection or network failure occurs. The event is then reported to the application via Global Call Alarm Management System (GCAMS). Currently, when an alarm is generated, the specific board on which the alarm occurred is provided to the application, but the specific port is not identified. If failures occur on both ports, two alarms are generated; however, if the failure condition is corrected on both ports, only one event is generated to notify the application that the alarm condition no longer exists. Also, no LAN disconnect network alarms are supported on 1000BT ports.

With this feature, the application is notified via a Global Call event whenever either of the 100BT or 1000BT Ethernet ports are connected or disconnected; that is, if both ports are disconnected, two events are generated, and if both are reconnected two additional events are generated.

To support this feature a new eQoSType value has been added to the IPM_QOS_ALARM_DATA data structure:

- QOSTYPE_NETWORKFAILURE1 - signifies port 1 fails/recovers (i.e., for 100BT Port 1 or 1000BT Port 1) (Dialogic® IPT Boards only)

The existing value of QOSTYPE_NETWORKFAILURE is used when port 0 fails/recovers (i.e., for 100BT Port 0 or 1000BT Port 0).

To enable the Global Call application to receive media network failure alarm events, follow the same steps provided in the *Dialogic® Global Call IP Technology Guide,* which include that the user enables the EVT_NETWORKFAILURE event for the board device using the **ipm_EnableEvent( )**. No new parameter has been added to enable the new port 1 events indicated above.

Note: Port redundancy is not supported; that is, even with failover activated (via the “failover” parameter) calls are not automatically moved from the primary port to the backup port.

1.25.2 Supported Boards

The following boards support this feature:

- Dialogic® IPT Boards

1.25.3 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about the Dialogic® IP Media Library API, see the *Dialogic® IP Media Library API Library Reference*.

1.26 Enhancement to its_sysinfo Tool

The its_sysinfo tool collects data from the system on which you execute it and provides information about the system environment: the operating system, computer architecture, System Release software, and operational logs.

With the Service Update, the *its_sysinfo.htm* file now includes a Windows Package Info section at the beginning of the file. For example:

```
                              WindowsPackageInfo

Active System Release

Dialogic(R) System Release 6.1 CompactPCI for Windows Build 125 (System Release)

Build Type: System Release
Install Location: C:\Program Files\Dialogic
Install Date: 2-20-2007 at 15:52:30
Installed By: Computing Customer

Installed Features
     Devel
     Runtime

Previously Installed System Release

Dialogic(R) System Release 6.1 CompactPCI for Windows Build 123 (System Release)

Build Type: System Release
Install Location: C:\Program Files\Dialogic
Install Date: 1-15-2007 at 15:29:40
Installed By: Computing Customer

Installed Features
     Devel
     Runtime
```

For more information about the its_sysinfo tool, refer to the *Dialogic® System Software Diagnostics Guide*.

1.27 Modify an Existing SIP Call Using re-INVITE for Dialogic® IPT Boards

With the Service Update, the user will be able to modify an existing SIP call using re-INVITE for the Dialogic® IPT Boards.

1.27.1 Feature Description

The ability to modify an existing SIP call using re-INVITE is now supported on Dialogic® IPT Boards. As with the Dialogic® DM/IP Boards, the Dialogic® IPT Boards do not currently support coder changes. Any request to change the coder or any of the coder properties (except direction) *must* be rejected by the application.

Note: The application cannot change the RTCP port. The RTCP port is always RTP port + 1 and the RTCP address is always equal to the RTP IP address. If the RTP port/IP address is changed as a result of a re-INVITE, the RTCP port changes accordingly.

1.27.2 Supported Boards

The following boards support this feature:

- Dialogic® IPT Boards

1.27.3 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about modifying an existing SIP call using re-INVITE, see the *Dialogic® Global Call IP Technology Guide*.

1.28 PDK Log File Detection

With the Service Update, the user is notified when a *pdk.cfg* file is missing (for example, deleted by mistake). The user has the option to view messages in an application like the Windows® Event Viewer and/or the RTF logs.

1.28.1 Feature Description

This feature troubleshoots a missing *pdk.cfg* file to detect and regenerate it. If the *pdk.cfg* file cannot be regenerated, a message displays in the Windows® Event Viewer and/or the RTF log.

PDK log messages are categorized as error, warning, or info messages. All error, warning, and info messages are displayed in the RTF log. Error and warning messages are also displayed in the Windows® Event Viewer because their occurrence indicates that the PDK download process has been critically impacted.

The following are examples of the Warning text that is displayed when a *pdk.cfg* file is detected and regenerated:

In the Windows® Event Viewer:

Date:	9/22/2006	Source:	PDKManager
Time:	2:22.21 PM	Category:	(1)
Type:	Warning	Event ID:	2
User:	N/A		
Computer:	MYCOMPUTER		

Description:
Category=1 PDFManager WARNING: PDK configured for use, but no pdk.cfg file found - regenerated pdk.cfg

In the RTF log:

```
09/22/2006 14:22:21.362   2320        2324 PDKManager             Warning      PDK configured
for use, but no pdk.cfg file found - regenerated pdk.cfg
```

The following is a example of the Error text that is displayed when PDKManager could not find the given FCD file:

In the Windows® Event Viewer:

Date:	9/22/2006	Source:	PDKManager
Time:	2:35:12 PM	Category:	(1)
Type:	Error	Event ID:	1
User:	N/A		
Computer:	MYCOMPUTER		

Description:
Category=1 PDFManager ERROR: Board: 1 Msg = Error, could not find fcd file ml1b_qsa_r2mf.fd

In the RTF log:

```
09/22/2006 14:42:34.206   3396        3316 PDKManager             Error        Board: 1  Msg =
Error, could not find fcd file ml1b_qsa_r2mf.fd.
```

1.28.2 Supported Boards

The following boards support this feature:

- Dialogic® DM/V-A, DM/V-B, and DM/V Media Boards

1.28.3 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about RTF, see the *Dialogic® System Software Diagnostics Guide*.

1.29 Media Channel Reset Capability (Stuck IP Media Channel Recovery)

With the Service Update, whenever an IP media channel gets into a “stuck” state, there is a way to recover that channel without having to redownload the board or restart the application.

A stuck channel is defined as a failure where the host application is unable to recover the channel and no further media operations are possible on that channel until the board is redownloaded or the application is restarted.

1.29.1 Feature Description

In high-density applications, media channels sometimes become stuck and no further processing can take place until the board is redownloaded or (in some cases) until the application is restarted.

This feature provides a new API function in the Dialogic® IP Media Library that enables the application to recover from the stuck channel and return it to an idle and usable state.

Note: Not all stuck channels are recoverable. Also, not all errors are stuck channel errors. See Section 1.29.4, “Feature Limitations or Restrictions”, on page 83 for more information.

1.29.2 Supported Boards

The following boards support this feature:

- Dialogic® DM/IP Boards

1.29.3 New IP Media Library API

The new Dialogic® IP Media Library API is **ipm_ResetChannel( )**. Call this API to initiate the reset of all media related components in the firmware and host library, returning them to the idle state. By default, resetting a channel results in the configuration parameters returning to default values (that is, parameter values return to the values assigned at download time). Any values that were customized after download have to be re-configured after the channel is reset. The **ipm_ResetChannel( )** function has an option to reset only the media channel state, leaving all parameters unaltered. The

ipm_ResetChannel( ) function only resets the media channel and does not have any effect on the signaling channel.

Modified Dialogic® Global Call API Functionality

The **gc_ResetLineDev( )** function has been modified to perform **ipm_ResetChannel( )** functionality in the background. However, when the **gc_ResetLineDev( )** function is issued in asynchronous mode, the application does not receive an **ipm_ResetChannel( )** termination event (IPMEV_RESET or IPMEV_RESET_FAIL), but receives a **gc_ResetLineDev( )** termination event as before.

Function reference information is provided next.

ipm_ResetChannel( )

Name: int ipm_ResetChannel(nDeviceHandle, eResetMode, usMode)

Inputs:

int nDeviceHandle	• IP Media device handle
eIPM_RESET_MODE eResetMode	• reset mode
unsigned short usMode	• async or sync mode setting

Returns: 0 on success
-1 on failure

Includes: srllib.h
ipmlib.h

Category: System Control

Mode: asynchronous or synchronous

Platform: DM/IP

■ Description

The **ipm_ResetChannel( )** function resets an IP Media channel.

Parameter	Description
nDeviceHandle	handle of the IP media device
eResetMode	specifies the reset mode ***Note:*** Only one value can be set at a time. The eIPM_RESET_MODE data type is an enumeration that defines the following values: • RESET_CHANNEL_ALL - Reset IP Media channel to its initial state after download. This includes resetting IP Media channel parameters to their values after download. • RESET_CHANNEL_STATE - Reset IP Media channel state to IDLE. IP Media channel parameter values will be unaffected when this reset mode is specified.
usMode	operation mode Set to EV_ASYNC for asynchronous execution or to EV_SYNC for synchronous execution.

The **ipm_ResetChannel( )** function supports both synchronous and asynchronous operation modes:

- In synchronous mode, successful completion of the **ipm_ResetChannel( )** function is indicated by a return value of 0. Operation failure is indicated by a return value of -1.
- In asynchronous mode, the **ipm_ResetChannel( )** function returns 0 to indicate that the operation was initiated successfully. This is supplemented by the delivery of an IPMEV_RESET event on successful completion of the operation. Failure to initiate the reset operation initiation is indicated when the **ipm_ResetChannel( )** function returns -1. If the

ipm_ResetChannel( ) function successfully initiates the reset operation asynchronously, but subsequently fails to complete successfully, an IPMEV_RESETFAIL event is delivered.

Termination Events

IPMEV_RESET
Indicates successful completion; that is, the supplied IP Media channel has been reset.

IPMEV_RESETFAIL
Indicates that the reset operation failed.

Cautions

- The **ipm_ResetChannel( )** function is a blocking call and, depending on the mode it is called in, does not allow any other function on that channel or thread.
- This reset functionality is only intended to be used on channels that are stuck and not responding. The function should not be used in place of **ipm_stop( )** and should only be called if events (with the exception of OFFERED) are not received within 30 seconds (SIP time-out) of when they are expected or if the application receives a taskfail. Overuse of the function creates unnecessary overhead and could affect system performance.

Errors

If the function returns -1 to indicate failure, call **ATDV_LASTERR( )** and **ATDV_ERRMSGP( )** to return one of the following errors:

EIPM_INTERNAL
Internal error

EIPM_SYSTEM
System error

EIPM_FWERROR
Firmware error

EIPM_BADPARM
Invalid parameter

EIPM_INV_MODE
Invalid mode

Example

```
#include <stdio.h>
#include <srllib.h>
#include <ipmlib.h>

typedef long int(*HDLR)(unsigned long);
void CheckResetEvent();

void main()
{
   int nDeviceHandle;

   // Register event handler function with srl
   sr_enbhdlr(EV_ANYDEV, IPMEV_RESET, (HDLR)CheckResetEvent);
   sr_enbhdlr(EV_ANYDEV, IPMEV_RESET_FAIL, (HDLR)CheckResetEvent);
```

```
   /*
   .
   .
   Main Processing
   .
   .
   */

   /*
   Application needs to issue reset on IP channel device handle, nDeviceHandle.
   ASSUMPTION: A valid nDeviceHandle was obtained from prior call to ipm_Open()
   and session has been started by calling ipm_StartMedia() some time earlier.
   ipm_Stop() has been issued on the device and stop completion reply does not
   come within acceptable time duration.
   */
   if (ipm_ResetChannel(nDeviceHandle, RESET_CHANNEL_STATE, EV_ASYNC) == -1)
   {
      printf("ipm_ResetChannel failed for device name = %s with error = %d\n",
             ATDV_NAMEP(nDeviceHandle), ATDV_LASTERR(nDeviceHandle));
      /*
      .
      .
      Perform Error Processing
      .
      .
      */
   }

   /*
   .
   .
   Continue Processing
   .
   .
   */
}

void CheckResetEvent()
{
   int nEventType = sr_getevttype();
   int nDeviceID  = sr_getevtdev();

   switch(nEventType)
   {
      /*
      .
      .
      List of Expected Events
      .
      .
      */

      /* Successful reply from ipm_ResetChannel() */
      case IPMEV_RESET:
         printf("Received IPMEV_RESET for device = %s",
                ATDV_NAMEP(nDeviceID));
         break;

      /* Failure reply from ipm_ResetChannel() */
      case IPMEV_RESET_FAIL"
         printf("Received IPMEV_RESET_FAIL for device = %s",
                ATDV_NAMEP(nDeviceID));
         break;
```

```
        default:
           printf("Received unexpected event = %d for device = %s\n",
                   nEventType, ATDV_NAMEP(nDeviceID));
           break;
     }
}
```

■ **See Also**

- **ipm_Stop( )**
- **gc_ResetLineDev( )**

1.29.4 Feature Limitations or Restrictions

The following restrictions and limitations apply to the feature:

- This solution only addresses scenarios where the firmware and host have lost synchronization or entered a bad state. DSP crashes or catastrophic firmware failures are not recoverable without re-download of the board firmware.
- In cases where host CPU utilization is approaching 100%, there is no guarantee that the **ipm_ResetChannel( )** function is propagated to the firmware in a timely manner. The assumption is that the application does not attempt to recover the channel until the CPU usage drops to a reasonable level (<70%).

1.29.5 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about the Dialogic® IP Media API, see the following documents:

- *Dialogic® IP Media Library API Programming Guide*
- *Dialogic® IP Media Library API Library Reference*

For more information about the Dialogic® Global Call API, see the following documents:

- *Dialogic® Global Call IP Technology Guide*
- *Dialogic® Global Call API Library Reference*

1.30 Dialogic® Global Call API Access to New H.323/Q.931 Message IEs

With the Service Update, you can access and configure the called and calling party number information elements (IEs) and various subfields of the H.322/Q.931 SETUP message with the Dialogic® IPT Boards.

1.30.1 Feature Description

You now have the ability to access additional fields in the calling party number (CGPN) and called party number (CDPN) IEs within a H.225/Q.931 SETUP message when using H.323 IP call signaling with the Dialogic® IPT Boards.

The SETUP message is a standard call signaling message used by a calling H.323 entity to establish a connection with the called entity. This message is currently supported with limited access to information fields in the CGPN and CDPN IEs. Presently, only the Called/Calling Party number can be modified by the application via **gc_SetUserInfo( )** or when invoking the **gc_MakeCall( )** function. You now can use an existing parameter set ID and its new parameter IDs to send and receive these CPN fields via Global Call over an IP network.

1.30.2 New Parameter IDs

An existing parameter set ID (IPSET_CALLINFO) and new parameter IDs support the CPN information as shown in the table below:

IPSET_CALLINFO Parameter IDs

Parameter ID	Set	Sent	Retrieve
IPPARM_CGPN_TYPE_OF_NUMBER IPPARM_CDPN_TYPE_OF_NUMBER IPPARM_CGPN_NUMBERING_PLAN_ID IPPARM_CDPN_NUMBERING_PLAN_ID IPPARM_CGPN_SCREENING_INDICATOR IPPARM_CGPN_PRESENTATION_INDICATOR	GC_PARM_BLK **gc_SetUserInfo( )**	**gc_MakeCall( )**	**gc_Extension( )** (IPEXTID_GETINFO) and asynchronous GCEV_EXTENTIONCMPLT completion event

Parameter ID Details

Parameter ID	Data Type & Size	Description
IPPARM_CGPN_TYPE_OF_NUMBER IPPARM_CDPN_TYPE_OF_NUMBER	Type: unsigned char Size: 1 byte	Contains the type of number in the CGPN or CDPN
IPPARM_CGPN_NUMBERING_PLAN_ID IPPARM_CDPN_NUMBERING_PLAN_ID	Type: unsigned char Size: 1 byte	Contains the numbering plan identification in the CGPN or CDPN
IPPARM_CGPN_SCREENING_INDICATOR	Type: unsigned char Size: 1 byte	Contains the screening indicator in the CGPN
IPPARM_CGPN_PRESENTATION_INDICATOR	Type: unsigned char Size: 1 byte	Contains the presentation indicator in the CGPN

The following definitions are in the *gcip_defs.h* file:

```
#define IPPARM_ CGPN_TYPE_OF_NUMBER              0x13
#define IPPARM_ CDPN_TYPE_OF_NUMBER              0x14
#define IPPARM_ CGPN_NUMBERING_PLAN_ID           0x15
#define IPPARM_ CDPN_NUMBERING_PLAN_ID           0x16
#define IPPARM_ CGPN_SCREENING_INDICATOR         0x17
#define IPPARM_ CGPN_PRESENTATION_INDICATOR      0x18
```

The following data variables are in the *gcip.h* file:

```
typedef unsigned char   CPN_TON;    /* Type of number */
typedef unsigned char   CPN_NPI;    /* Numbering plan identification */
typedef unsigned char   CPN_SI;     /* Screening Indicator      */
typedef unsigned char   CPN_PI;     /* Presentation Indicator */
```

1.30.3 Enabling the Setting and Retrieving of Q.931 Message IEs

To enable the setting and retrieving of all supported Q.931 message IEs, set the h323_msginfo_mask field in the IP_VIRTBOARD structure to a value of IP_H323_MSGINFO_ENABLE for each Dialogic® IPT Board device **before** calling **gc_Start( ).**

Note: By default, the underlying H.323 stack is not enabled to receive incoming Q.931 message IEs.

A code snippet showing how to do this is given below:

```
IP_VIRTBOARD virtBoard[MAX_BOARDS];
memset(virtBoard,0,sizeof(IP_VIRTBOARD) * MAX_BOARDS);
bid = 1;
INIT_IP_VIRTBOARD(&virtBoard[bid]);
// fill up other board parameters
….
 virtBoard[bid].localIP.ip_ver = IPVER4;
 virtBoard[bid].localIP.u_ipaddr.ipv4 = (unsigned int) IP_CFG_DEFAULT;
…
virtBoard[1].h323_msginfo_mask = IP_H323_MSGINFO_ENABLE;
//Then use the virtBoard structure in the gc_Start() function appropriately
```

You can also enable reception of other H.323 fields simultaneously via the code:

```
virtBoard[1].h323_msginfo_mask = IP_H323_MSGINFO_ENABLE | IP_H323_ANNEXMMSG_ENABLE;
```

1.30.3.1 Stopping the Reception of CPN Information

Currently, there is no way for the user application to turn off the reception of Q.931 IEs received by the underlying H323 stack once they are enabled, without stopping the application or restarting the stack.

1.30.4 Setting Up CPN Fields in the GC_PARM_BLK Data Structure

Before calling the **gc_MakeCall( )** function, you must set up the CPN fields to be included in the GC_PARM_BLK data structure. The GC_PARM_BLK should include the existing parameter set ID IPSET_CALLINFO and the newly defined parameter IDs described in the New Parameter IDs section, which specifies which CPN fields are to be set in the parameter block structure of a Make Call block.

To set up the CPN fields in the GC_PARM_BLK structure, call the **gc_util_insert_parm_ref( )** function.

Code Example

The following is an example of how to specify the CPN fields for sending.

```
#include <stdio.h>
#include <string.h>
#include <gcip.h>
#include <.h>


void main()
{
   CPN_TON    cgpn_ton, cdpn_ton;
   CPN_NPIcgpn_npi, cdpn_npi;
   CPN_SIcgpn_si;
   CPN_PIcgpn_pi;
   GC_PARM_BLKPpParmBlock;

  /*. .   Main Processing….*/
  /*  Set CPN fields in the Make Call Block to be sent out via SETUP message */

  cgpn_ton = 0x1;   // Note that the field values must be valid.
  cdpn_ton = 0x4;
  cgpn_npi = 0x1;
  cdpn_npi = 0x1;
  cgpn_si = 0x0;
  cgpn_pi = 0x0;

    gc_util_insert_parm_ref(&pParmBlock,
                            IPSET_ CALLINFO,
                            IPPARM_CGPN_TYPE_OF_NUMBER,
                            sizeof(unsigned char),
                            &cgpn_ton);

    gc_util_insert_parm_ref(&pParmBlock,
                            IPSET_ CALLINFO,
                            IPPARM_CDPN_TYPE_OF_NUMBER,
                            sizeof(unsigned char),
                            &cdpn_ton);

    gc_util_insert_parm_ref(&pParmBlock,
                            IPSET_ CALLINFO,
                            IPPARM_CGPN_NUMBERING_PLAN_ID,
                            sizeof(unsigned char),
                            &cgpn_npi);
```

```
    gc_util_insert_parm_ref(&pParmBlock,
                            IPSET_ CALLINFO,
                            IPPARM_CDPN_NUMBERING_PLAN_ID,
                            sizeof(unsigned char),
                            &cdpn_npi);

    gc_util_insert_parm_ref(&pParmBlock,
                            IPSET_ CALLINFO,
                            IPPARM_CGPN_SCREENING_INDICATOR,
                            sizeof(unsigned char),
                            &cgpn_si);

    gc_util_insert_parm_ref(&pParmBlock,
                            IPSET_ CALLINFO,
                            IPPARM_CGPN_PRESENTATION_INDICATOR,
                            &cgpn_pi);

  /*. .. Continue Main processing. … call gc_MakeCall() */
}
```

1.30.5 Generating CPN Fields in a SETUP Message

If you choose to insert the CPN data into the GC_MAKECALL_BLK using the **gc_SetUserInfo( )** function, refer to the *Dialogic® Global Call IP Technology Guide* for details on how this can be done.

After setting the CPN signaling message fields as described in the Setting Up CPN Fields in the GC_PARM_BLK Data Structure section, the user application calls the **gc_MakeCall( )** function, passing it a pointer to the GC_MAKECALL_BLK.

1.30.6 Retrieving CPN Information

When the underlying H.323 stack is enabled to retrieve incoming call info fields, including CPN information, any CPN fields detected in an associated H.225 message will be retrieved and stored in Global Call.

Use the **gc_Extension( )** function to retrieve the CPN data within any valid incoming H.225 message containing CPN fields, while a call is in any state, after the OFFERED state. This is similar to receiving other call related information via the GCEV_EXTENSIONCMPLT event received as a termination event to the **gc_Extension( )** function.

Set the target_type to GCTGT_GCLIB_CRN. Set the target_id to the actual CRN. If incoming CPN data is available, the information is included with the corresponding GCEV_EXTENSIONCMPLT asynchronous termination event.

The extevtdatap field in the METAEVENT structure for the GCEV_EXTENSIONCMPLT event is a pointer to an EXTENSIONEVTBLK structure that contains a GC_PARM_BLK with the requested CPN information.

When trying to retrieve the CPN information, it is necessary to specify each of the CPN parameters in the extension request, for which information from the stack is needed.

Code Examples

Specifying CPN Field for Receiving

A code example of how to specify the CPN fields for receiving is shown below. This example is just for the Calling Number Type of Number field. The method to specify the other CPN data would be similar.

```
int getCPNInfo(CRN crn)
{
   GC_PARM_BLKP gcParmBlk = NULL;
   GC_PARM_BLKP retParmBlk;
   int frc;

   frc = gc_util_insert_parm_val(&gcParmBlk,
                                 IPSET_CALLINFO,
                                 IPPARM_CGPN_TYPE_OF_NUMBER,
                                 sizeof(unsigned char),1);

   if (GC_SUCCESS != frc)
   {
       return GC_ERROR;
   }

   frc = gc_Extension (GCTGT_GCLIB_CRN,
                       crn,
                       IPEXTID_GETINFO,
                       gcParmBlk,
                       &retParmBlk,
                       EV_ASYNC);
   if (GC_SUCCESS != frc)
   {
       return GC_ERROR;
   }

   gc_util_delete_parm_blk(gcParmBlk);

    return GC_SUCCESS;

}
```

Retrieving CPN Information

A code example of how to extract CPN information from an unsolicited GCEV_EXTENSIONCMPLT event received as a result of a request for call-related information is shown below:

```
int OnExtension(GC_PARM_BLKP parm_blk,CRN crn)
{
   GC_PARM_DATA *parmp = NULL;
   parmp = gc_util_next_parm(parm_blk,parmp);

   if (!parmp)
   {
      return GC_ERROR;
   }

   while (NULL != parmp)
   {
     switch (parmp->set_ID)
     {
       case IPSET_CALLINFO:
```

```
            switch (parmp->parm_ID)
            {
              case IPPARM_CGPN_TYPE_OF_NUMBER:
                    printf("\tReceived CPN data Calling Party Type of Number: %d\n", (*(unsigned
char*)(parmp->value_buf)));
              break;

              case IPPARM_CDPN_TYPE_OF_NUMBER:
                    printf("\tReceived CPN data Called Party Type of Number: %d\n", (*(unsigned
char*)(parmp->value_buf)));
              break;

              default:
                    printf("\tReceived unknown extension parmID %d\n",
                    parmp->parm_ID);

              break;
            }
   break;
     }
   parmp = gc_util_next_parm(parm_blk,parmp);
   }
}
```

1.30.7 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about the Dialogic® Global Call IP, see the following documents:

- *Dialogic® Global Call IP Technology Guide*
- *Dialogic® Global Call API Programming Guide*
- *Dialogic® Global Call API Library Reference*

For more information about the Dialogic® IP Media API, see the following documents:

- *Dialogic® IP Media Library API Programming Guide*
- *Dialogic® IP Media Library API Library Reference*

1.31 On-Demand Full Reset of Dialogic® DM3 Boards

With the Service Update, the user has the ability to perform a full Power On Self Test (POST) through the Dialogic® Configuration Manager (DCM) as well as with the Dialogic® NCM API (**NCM_SetValueEx( )** and **NCM_GetValueEx( )** functions) prior to downloading the Dialogic® DM3 Board.

1.31.1 Feature Description

Users have the option to perform a full POST prior to downloading the Dialogic® DM3 Boards to ensure that the boards start up in a known state in case of a possible failover (which may occasionally cause a bus reset issue). To choose a full POST option, select a

new parameter, **Run_full_POST_at_download**, on the Misc property sheet in the DCM. For NCM users, a new parameter can be used with the **NCM_SetValueEx( )** and **NCM_GetValueEx( )** functions.

The values for the parameter are as follows:

- True: enables full POST to be run before board download.
- False [default]: disables the full POST feature for Dialogic® DM3 Boards.

The user can select a single board. However, running full POST prior to board download will add about 45 seconds for Dialogic® DM/V1200A-4E1 Boards, about 50 seconds for Dialogic® DM/IP601 Boards, and about 50 seconds for Dialogic® DM/V1200BTEC Boards to the overall startup process no matter how many boards are selected. For example, for Dialogic® DM/V-A Boards, the additional time would be about 45 seconds if the user selects one board or all boards for a full POST before download.

1.31.2 Supported Boards

The following boards support this feature:

- Dialogic® DM/V1200A-4E1 Media Boards
- Dialogic® DM/IP601-100cPCI IP Boards
- Dialogic® DM/V1200BTEC Media Boards

1.31.3 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about the Dialogic® NCM API, see the following documents:

- *Dialogic® Native Configuration Manager API Library Reference*
- *Dialogic® Native Configuration Manager API Programming Guide*

For additional information about the Dialogic® Configuration Manager (DCM), see the following:

- *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide*

1.32 New Dialogic® Diagnostics Management Console

The Service Update introduces the Dialogic® Diagnostics Management Console (DMC) version 1.0. This GUI tool provides a means of quickly launching Dialogic® diagnostic utilities and viewing various log files created with those utilities.

The DMC:

- Provides a single portal for launching diagnostic tools:
  - AppMon
  - Castrace
  - Isdntrace
  - Dlgsnapshot
  - Dm3post
  - Debugangel
  - Getver
  - its_sysinfo
  - Pdktrace
  - Pstndiag
  - RTF Manager
  - StatusMon
- Supports local and remote execution of tools. Diagnostic tools are launched remotely via the standard remote control methods provided with the operating system, such as SSH or Remote Desktop.
- Lists the diagnostic logs available both locally and remotely for viewing.
- Launches appropriate viewers for displaying logged data.

Note: Java Runtime Environment (JRE) version 1.5 must be installed on your system in order to run the DMC.

For more information, refer to the *Dialogic® System Software Diagnostics Guide*. The DMC also has online help.

1.33 New Runtime Trace Facility (RTF) Manager

The Service Update introduces the RTF Manager, a new GUI for the Runtime Trace Facility (RTF) diagnostic tool. RTF Manager allows users to easily configure logging and tracing levels. Previously, users had to manually edit the RTF configuration file.

Note: Java Runtime Environment (JRE) version 1.5 must be installed on your system in order to run the RTF Manager.

For more information about the RTF Manager, refer to the *Dialogic® System Software Diagnostics Guide*.

1.34 Support for Reporting Billing Type

With this Service Update, for Dialogic® DM3 Boards, there is now a way for the application to know which billing type (for a call on PDK R2 protocol) was received when the lines are available for call establishment. B tones are sent to indicate whether the line is available,

and also to indicate the type of billing for the call (for example, CHARGE, NO CHARGE, or CHARGE WITH CLEARING FROM INBOUND).

This feature is already supported on Dialogic® Springware Boards; however, CHARGE WITH CLEARING FROM INBOUND is a new billing type that is also supported on Springware Boards now.

For further information about this feature, see the description of the **gc_GetCallInfo( )** function CALLINFOTYPE info_id parameter in the *Dialogic® Global Call E1/T1 CAS/R2 Technology Guide*.

1.35 Runtime Control of Double Answer for R2MF

With this Service Update, a connection method called double answer is now supported for rejecting collect calls on a call-by-call basis. For further information about this feature, see the *Dialogic® Global Call E1/T1 CAS/R2 Technology Guide*.

1.36 Additional Supported Operating System Security Updates

With the Service Update, the following Windows® Operating System security updates are now supported:

- Windows® 2003 Server R2
- Windows® 2000 Update Rollup 1 for Service Pack 4

1.37 Media Channel Reset Capability (Stuck Channel Recovery)

With the Service Update, whenever a media channel gets into a “stuck” state, there is a way to recover that channel without having to redownload the board or restart the application.

Note: A stuck channel is defined as a failure where the host application is unable to recover the channel and no further media operations are possible on that channel until the board is redownloaded or (in some cases) the application is restarted.

1.37.1 Feature Description

In high-density applications, media channels sometimes become stuck and no further processing can take place until the board is redownloaded or (in some cases) until the application is restarted.

This feature provides new API functions in the Dialogic® Voice library and in the Dialogic® Continuous Speech Processing (CSP) library that enable the application to recover from the stuck channel and return it to an idle and usable state.

Note: Not all stuck channels are recoverable. Also, not all errors are stuck channel errors. See Section 1.37.2, “Restrictions and Limitations”, on page 100 for more information.

Supported Boards

All media span boards support this media channel reset feature, namely Dialogic® DM/V, DM/V-A, DM/V-B, and DM/IP Boards.

New APIs

The two new API functions are as follows:

- **dx_resetch( )** - Call this API to recover the media channel when the channel is stuck and in a recoverable state. If the channel is recovered, a TDX_RESET event is generated to the application, which enables the application to reuse the channel for more media functions. If the channel is not in a recoverable state, a TDX_RESETERR event is sent back to the application indicating that the specific channel is not recoverable.
- **ec_resetch( )** - Call this API to recover the CSP channel when the channel is stuck and in a recoverable state. If the channel is recovered, TDX_RESET and TEC_RESET events are generated to the application, which enables the application to reuse the channel for more media functions. If the channel is not in a recoverable state, TDX_RESETERR and TEC_RESETERR events are sent back to the application indicating that the specific channel is not recoverable. Note that the **ec_resetch( )** function resets both the voice and the CSP channels.

Function reference information is provided next.

dx_resetch( )

Name: dx_resetch (chdev, mode)

Inputs: int chdev
- valid channel device handle

int mode
- mode of operation

Returns: 0 if success
-1 if failure

Includes: srllib.h
dxxxlib.h

Category: I/O

Mode: asynchronous or synchronous

Platform: DM3

Description

The **dx_resetch( )** function recovers a channel that is "stuck" (busy or hung) and in a recoverable state, and brings it to an idle and usable state. This function blocks all other functions from operating on the channel until the function completes.

Parameter	Description
chdev	Specifies the valid device handle obtained when the channel was opened using **dx_open( )**
mode	Specifies the mode of operation: • EV_ASYNC – asynchronous mode. The calling thread returns immediately so it can process media functionality on other channels. • EV_SYNC – synchronous mode. The calling thread waits until the channel is recovered or discovers that the channel is not in a recoverable state.

In synchronous mode, 0 is returned if the function completes successfully, and -1 is returned in case of error.

In asynchronous mode, the TDX_RESET event is generated to indicate that the channel was recovered and is in an idle and usable state. The TDX_RESETERR event is generated to indicate that the channel is not recoverable. Issuing any other media calls on this channel will result in an error.

Cautions

- The **dx_resetch( )** function is intended for use on channels that are stuck and not responding. Do **not** use it in place of **dx_stopch( )**. Use **dx_resetch( )** only if you do not receive an event within 30 seconds of when it's expected. Overuse of this function creates unnecessary overhead and may affect system performance.

Errors

If the function returns -1, use the Dialogic® Standard Runtime Library (SRL) Standard Attribute function **ATDV_LASTERR( )** to obtain the error code or use **ATDV_ERRMSGP( )** to obtain a descriptive error message. One of the following error codes may be returned:

EDX_BADPARM
: Invalid parameter

EDX_FWERROR
: Firmware error

EDX_NOERROR
: No error

Example

```
#include <srllib.h>
#include <dxxxlib.h>

main()
{
   int chdev, srlmode;
   /* Set SRL to run in polled mode. */
   srlmode = SR_POLLMODE;

   if (sr_setparm(SRL_DEVICE, SR_MODEID, (void *)&srlmode) == -1) {
   /* process error */
   }

   /* Open the channel using dx_open( ). Get channel device descriptor in
   * chdev.
   */

   if ((chdev = dx_open("dxxxB1C1",NULL)) == -1) {
   /* process error */
   }

   /* continue processing */
   . .
   /* Force the channel to idle state. The I/O function that the channel
    * is executing will be terminated, and control passed to the handler
    * function previously enabled, using sr_enbhdlr(), for the
    * termination event corresponding to that I/O function.
    * In asynchronous mode, dx_stopch() returns immediately,
    * without waiting for the channel to go idle.
    */

   if ( dx_stopch(chdev, EV_ASYNC) == -1) {
   /* process error */
   }

   /* Wait for dx_stopch() to stop the channel and return the termination event
    * for the present media function.
    */

   /* After waiting for 30 secs if the termination event is not returned, issue a
    * dx_resetch() to reset the channel.
    */

   if (dx_resetch(chdev, EV_ASYNC) <0 )
   {
      /*process error */
```

```
    }
    /* Wait for TDX_RESET or TDX_RESETERR events */
}
```

See Also

- **ec_resetch( )** in the *Dialogic® Continuous Speech Processing API Library Reference*

ec_resetch( )

Name: ec_resetch (chdev, mode)

Inputs: int chdev — • valid channel device handle

int mode — • mode of operation

Returns: 0 if success
-1 if failure

Includes: srllib.h
eclib.h

Category: I/O

Mode: asynchronous or synchronous

Platform: DM3

■ Description

The **ec_resetch( )** function recovers a channel that is "stuck" (busy or hung) and in a recoverable state, and brings it to an idle and usable state. This function blocks all other functions from operating on the channel until the function completes. This function recovers both the CSP channel and the voice channel.

Parameter	Description
chdev	Specifies the valid device handle obtained when the channel was opened using **dx_open( )**
mode	Specifies the mode of operation: • EV_ASYNC – asynchronous mode. The calling thread returns immediately so it can process media functionality on other channels. • EV_SYNC – synchronous mode. The calling thread waits until the channel is recovered or discovers that the channel is not in a recoverable state.

In synchronous mode, 0 is returned if the function completes successfully, and -1 is returned in case of error.

In asynchronous mode, the TDX_RESET and the TEC_RESET events are generated to indicate that the channel was recovered and is in an idle and usable state. The TDX_RESETERR and the TEC_RESETERR events are generated to indicate that the channel is not recoverable. Issuing any other media calls on this channel will result in an error.

■ Cautions

- The **ec_resetch( )** function is intended for use on channels that are stuck and not responding. Do **not** use it in place of **ec_stopch( )**. Use **ec_resetch( )** only if you do not receive an event within 30 seconds of when it's expected. Overuse of this function creates unnecessary overhead and may affect system performance.

Errors

If the function returns -1, use the Dialogic® Standard Runtime Library (SRL) Standard Attribute function **ATDV_LASTERR( )** to obtain the error code or use **ATDV_ERRMSGP( )** to obtain a descriptive error message. One of the following error codes may be returned:

EDX_BADPARM
: Invalid parameter

EDX_FWERROR
: Firmware error

EDX_NOERROR
: No error

Example

```
#include <stdio.h>
#include <srllib.h>
#include <dxxxlib.h>
#include <eclib.h>
#include <errno.h> /* include in Linux applications only; exclude in Windows */

main()
{
   int chdev, srlmode;
   /* Set SRL to run in polled mode. */
   srlmode = SR_POLLMODE;
   if (sr_setparm(SRL_DEVICE, SR_MODEID, (void *)&srlmode) == -1) {
   /* process error */
   }

   /* Open the channel using dx_open( ). Get channel device descriptor
   * in chdev.
   */
   if ((chdev = dx_open("dxxxB1C1",0)) == -1) {
   /* process error */
   }
   /* continue processing */
   . .
   /* Force the channel to idle state. The I/O function that the channel
   * is executing will be terminated, and control passed to the handler
   * function previously enabled, using sr_enbhdlr(), for the
   * termination event corresponding to that I/O function.
   * In the asynchronous mode, ec_stopch() returns immediately,
   * without waiting for the channel to go idle.
   */
   if (ec_stopch(chdev, FULLDUPLEX, EV_ASYNC) == -1) {
   /* process error */
   }

   /* Wait for the termination events (TEC_STREAM and/or TDX_PLAY) */

   /* After waiting for 30 secs, if the channel is still in a busy state,
    * issue ec_resetch() to reset both the CSP channel and the voice channel.
    * When issued in asynchronous mode, it will return both (TEC_RESET/TEC_RESETERR)
    * and (TDX_RESET/TDX_RESETERR) events.
    */

   if (ec_resetch(chdev, EV_ASYNC)  == -1 ) {
      /* process error */
   }
```

```
    /* Wait for TEC_RESET/TEC_RESETERR and TDX_RESET/TDX_RESETERR */
}
```

See Also

- **dx_resetch( )** in the *Dialogic® Voice API Library Reference*

Implementation Guidelines

The following guidelines apply when implementing the media channel reset capability using the Voice API:

- It is recommended that you issue the function in asynchronous mode for more efficient processing. In synchronous mode, the calling thread is blocked until the function completes, which may take up to a minute in worst-case scenarios.
- The **dx_resetch( )** function is intended for use on channels that are stuck and not responding. Do **not** use it in place of **dx_stopch( )**. Use **dx_resetch( )** only if you do not receive an event within 30 seconds of when it's expected. Overuse of this function creates unnecessary overhead and may affect system performance.
- If you call **dx_resetch( )** immediately following **dx_stopch( )** without waiting at least 30 seconds for **dx_stopch( )** to complete, you will not receive events, such as TDX_PLAY and TDX_RECORD, even if the stop operation is successful and the channel was not stuck. Instead, you will only receive the TDX_RESET event if the channel recovery is successful or the TDX_RESETERR event if the channel is not recoverable.
- If you call **dx_resetch( )** without first using **dx_stopch( )** to stop the channel, the voice library will internally call **dx_stopch( )** and wait 30 seconds for it to complete. If the internal stop channel is successful, you will receive the TDX_RESET event only. If the internal stop channel is unsuccessful, the voice library will then call **dx_resetch( )**. Once a reset is attempted, you will receive the TDX_RESET event if the channel recovery is successful or the TDX_RESETERR event if the channel is not recoverable.
- Unrecoverable channels are written to a log file in the DebugAngel tool or the Runtime Trace Facility (RTF) tool. See the *Dialogic® System Software Diagnostics Guide* for more information on these tools.

The following guidelines apply when implementing the media channel reset capability using the CSP API:

- The guidelines described for **dx_resetch( )** and **dx_stopch( )** apply to the **ec_resetch( )** and **ec_stopch( )** functions in the CSP API.
- For CSP applications, it is recommended that you use **ec_resetch( )** since this function resets both the voice and the CSP channels. The **dx_resetch( )** function resets the voice channels only.

1.37.2 Restrictions and Limitations

The following restrictions and limitations apply to the media channel reset feature:

- This feature only addresses scenarios where the firmware and the host library have lost synchronization or an event has not been propagated. DSP crashes, catastrophic firmware failures (killtasks), or unsynchronized firmware state machines are **not** recoverable without redownload of the board.

- This feature only addresses channels that become stuck while performing play and record, tone generation, or FSK operations. It also addresses channels that become stuck during CSP play or record operations.
- This feature does **not** address reset of IP media channels on Dialogic® DM/IP Boards. It only addresses the reset of voice channels on DM/IP Boards.
- The reset may not succeed if CPU utilization on the host system is close to 100 percent. It is recommended that the CPU usage be at a reasonable level (less than 70 percent) before you attempt a channel reset.

1.37.3 Documentation

The online bookshelf provided with Dialogic® System Release 6.1 CompactPCI for Windows® contains information about all system release features including features for application development, configuration, administration, and diagnostics.

For more information about the Dialogic® Voice API, see the following documents:

- *Dialogic® Voice API Programming Guide*
- *Dialogic® Voice API Library Reference*

For more information about the Dialogic® Continuous Speech Processing (CSP) API, see the following documents:

- *Dialogic® Continuous Speech Processing API Programming Guide*
- *Dialogic® Continuous Speech Processing API Library Reference*

1.38 Notification of Layer 1 Alarm Events on Dialogic® SS7 Boards

With the Service Update, the support for alarm notification has been added for Dialogic® SS7 Boards. By adding support for alarm notification, applications are able to better determine which devices are available for making and receiving calls, or enabling/disabling voice activity.

For further information about this feature, see the *Dialogic® Global Call SS7 Technology Guide*.

1.39 Dialogic® Global Call Support for Time Slots on Dialogic® SS7 Boards Running in DTI Mode

With the Service Update, the Dialogic® Global Call API works with Dialogic® SS7 Boards that include trunks not configured for SS7 signalling (DTI mode); i.e., all the time slots on these trunks operate in clear channel mode.

For further information about this feature, see the *Dialogic® Global Call SS7 Technology Guide*.

Release Issues

The table below lists issues that can affect the hardware and software supported in Dialogic® System Release 6.1 CompactPCI for Windows®. The following information is provided for each issue:

Issue Type

This classifies the type of release issue based on its effect on users and its disposition:

- Known – A minor hardware or software issue. This category includes interoperability issues (i.e., issues relating to combining different Dialogic® products in the same system) and compatibility issues (i.e., issues that affect the use of Dialogic® products in with third-party software or hardware). Known issues are still open but may or may not be fixed in the future.
- Known (permanent) – A known hardware or software issue or limitation that will not be fixed in the future.
- Resolved – A hardware or software issue that was resolved (usually either fixed or documented) in this release.

Defect No.

A unique identification number that is used to track each issue reported via a formal Change Control System. Additional information on defects may be available via the Defect Tracking tool at *http://membersresource.dialogic.com/defects/*.
Note that when you select this link, you will be asked to either LOGIN or JOIN.

PTR No.

Number from problem tracking system used prior to March 27, 2006. For customer convenience, both the PTR number and the corresponding defect number are shown. For issues reported after March 27, 2006, this column contains "--" and only the defect number is used to track the issue.

SU No.

For defects that were resolved in a Service Update, indicates the Service Update number. For defects that were resolved when the base release was generally available (before any Service Updates), a "--" is shown. For non-resolved issues, this information is left blank.

Product or Component

The product or component to which the issue relates, typically one of the following:

- A system-level component; for example, Host Admin
- A hardware product; for example, Dialogic® DM/V Media Boards
- A software product; for example, the Dialogic® Global Call library

Description

A summary description of the issue. For non-resolved issues, a workaround is included when available.

The following table lists all issues that relate to this release, sorted by Issue Type. For other sort orders, use the following links:

- View issues sorted by Service Update Number
- View issues sorted by Product or Component
- View issues sorted by Defect Number

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows®

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Known	IPY00034117	--		Call Control	When running on system application, there's TASKFAIL on IP call control.
Known	IPY00042226	--		Diagnostics	PDK Trace maximum log file size is only 100 megabytes when it should be unlimited. ***Workaround:*** To avoid losing data, use the new log file array option ("-a") with size > 1. Log files are binary format and rarely grow beyond this limit; however in the event this happens and upon reaching the limit, the log file is saved with the timestamp naming convention, and a new log file is created. See Section 1.12, "File Management Enhancements for PDK Trace Tool", on page 45 for more information
Known	IPY00006814	36138		Diagnostics	The ISDNtrace tool does not parse ISDN messages correctly. It appears to be parsing ISDN messages as if they were DPNSS/DASS2 messages.
Known	IPY00006136	35891		DM/V4800BC Boards	Heavy plays or records using the 11kHz 16-bit linear coder (176 kbps) above ~ 240 channels on a Dialogic® DM/V4800BC Board with ML1B might cause quality degradation on the audio due to silence gaps.
Known	IPY00093815	--		DM3 Voice	A Blue Screen of Death (BSOD) occurs during heavy memory usage with no notification to the application.
Known	IPY00079022	--		DM3 Voice	In the DV_TPT data structure, the TF_SETINIT flag for DX_MAXSIL termination condition has no effect on standard records (**dx_reciottdata( )** family) with Dialogic® DM3 Boards, and will be ignored.
Known	IPY00032276	35956		Global Call	The GCEV_EXTENSION event is missing when the pattern changes from 5 to 6 with gtone_16xt_16_r2mf.pcd.
Known	IPY00006777	36146		Host Admin	The DMT160TEC Board fails to download intermittently when DebugAngel is running. The ML2 ISDN load does not consistently download on DMT160TEC when the SBC has more than 2 GB of memory. ***Workaround:*** Disable DebugAngel during download.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Known	IPY00006148	36279		Hot Swap Kit	The procedure for uninstalling the Hot Swap Kit (HSK) and Redundant Host (RH) software is given in the *Dialogic® System Release 6.1 CompactPCI for Windows® Software Installation Guide*. If you do not follow the HSK/RH uninstall procedure and try to uninstall the HSK before uninstalling the RH software, you will encounter a problem. The following will occur: when you try to uninstall the HSK first, you will be instructed to uninstall RH first, then prompted to do so. RH will uninstall and the machine will reboot. After reboot, when you try to uninstall the HSK, the uninstall will fail due to a missing shared file. ***Workaround:*** Execute hsk.exe from the hsk directory on the install CD, which will prompt you to uninstall the HSK software. This will uninstall the HSK software without the "missing file" error.
Known				Hot Swap Kit	If there is a non-functional CompactPCI board in the system, other functional CompactPCI boards might not be detected by the operating system after switchover is performed using Redundant Host (RH).
Known	IPY00043963	--		IPT Boards	TASKFAIL on Dialogic® IPT2400C Board.
Known	IPY00036815	--		IPT Boards	1000BT and 100BT Port 1 cannot be set to primary port for handling RTP media.
Known (permanent)	IPY00010898	36117		DM/V4800BC Boards	Heavy simultaneous plays or records using the 8kHz 16-bit linear coder (128 kbps) above ~ 400 channels on a Dialogic® DM/V4800BC Board with ML1B might cause quality degradation on the audio.
Resolved	IPY00078854	--	179	Board Detection	When there are two Dialogic® IPT Boards installed, and a single board stop of Dialogic® services is being performed on one of the boards, the status indicators on both of the IPT Boards turn red in Dialogic® Configuration Manager (DCM). Only the board that is being stopped should have a red status indicator at that time; the other board should be green.
Resolved	IPY00045395	--	178	Board Detection	After performing a single board stop of Dialogic® services on a Dialogic® IPT Board and then physically removing the board from the system, Dialogic® Configuration Manager (DCM) generates warning messages. The board that was removed is still shown in the DCM tree with board status as running.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00044713	--	174	Board Detection	Dialogic® IPT Boards do not report the correct serial number in OA&M activities such as in Dialogic® Configuration Manager (DCM). **Note:** When using the Service Update where this fix is introduced, the IPT Board firmware must be updated by the user prior to board initialization. For information about updating IPT Board firmware, see IPT Board Firmware Update Utility in the Documentation Updates section of this Release Update.
Resolved	IPY00044686	--	174	Board Detection	**ncm_RestoreConfiguration( )** does not work with Dialogic® IPT Boards. An error is reported in the system event log during system startup or at run time if this API call is made. **Note:** When using the Service Update where this fix is introduced, the IPT Board firmware must be updated by the user prior to board initialization. For information about updating IPT Board firmware, see IPT Board Firmware Update Utility in the Documentation Updates section of this Release Update.
Resolved	IPY00039538	--	148	Board Detection	Error messages are seen in the Windows® event viewer indicating that the RTF server is not running. This is occurring because none of the Dialogic® services have a dependency configured on the RTF service.
Resolved	IPY00037356	--	128	Board Detection	DCM assigns the same physical slot ID to two boards (in different physical slots).
Resolved	IPY00033640	--	121	Board Detection	During hot insertion, error appears in DCM and DebugAngel for Dummy Board.
Resolved	IPY00033164	--	120	Board Detection	Lost Packet Alarm not working on Dialogic® IPT Boards.
Resolved	IPY00038074	--	139	Board Download	The OAMSYSLOG component reports multiple “DM3FDSP - GetOverlappedResult()[2] timeout for board 5, Error= 121” entries in RTF logs during load test.
Resolved	IPY00036025	--	121	Board Download	Resolve crashes during DM/V-A detection/download on Motorola cPCI chassis.
Resolved	IPY00033228	--	139	Board Download	Cannot route voice device if it is not on the same board as the digital frontend device.
Resolved	IPY00035350	--	122	Call Control	While sending NonStandard Control data in an H.323 message, if the input string contains a byte with value 0x00, all the data after this byte will not be sent.
Resolved	IPY00033393	--	111	Call Control	The channel state is being changed for non-blocking alarms.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00032900	--	106	Call Control	In async mode, detach doesn't work. In sync mode, GCEV_ATTACH fails after a detach is issued.
Resolved	IPY00092350	--	199	Conferencing	Memory allocation errors occur when using the conferencing APIs.
Resolved	IPY00039068	--	143	Conferencing	The **dcb_addtoconf( )** function returns failure, and ATDV_ERRMSGP shows the error message as "Timed out waiting for reply from firmware."
Resolved	IPY00036423	--	127	Conferencing	Sometimes a noise is generated when a party leaves a conference; the noise disappears when a party is added to the conference.
Resolved	IPY00028633	35748	127	Conferencing	Sometimes a noise is generated when a party leaves a conference; the noise disappears when a party is added to the conference.
Resolved	IPY00009499	33932	127	Conferencing	A loud scratch/click sound occurs when entering a conference when 1-2 parties are already in the conference.
Resolved	IPY00007470	32437	127	Conferencing	A sharp noise occurs when changing conference resource mode to MSPA_MODERECVONLY.
Resolved	IPY00006707	33803	127	Conferencing	Sometimes a noise is generated when a party leaves a conference; the noise disappears when a party is added to the conference.
Resolved	IPY00094129	--	199	Conferencing (DCB)	The **dcb_remfromconf( )** function fails to return an error when removing the last party in the conference.
Resolved	IPY00080914		201	Conferencing (DCB)	An error is received when calling **dcb_evtstatus( )** function.
Resolved	IPY00099304	--	201	Configuration	A Blue Screen of Death (BSOD) results with no specific pattern/error observed in the RTF logs when using a new chassis model with the Dialogic DMN160TEC and DMV4800BC boards.
Resolved	IPY00092546	--	201	Configuration	SRLGetSubDevicesOnVirtualBoard returns TYPE_R4_DCB_BOARD (514) for dcbB1D1 when it should return TYPE_R4_DCB_DSP (529).
Resolved	IPY00082064	--	195	Configuration	The Dialogic® Service fails to start with four Dialogic® DMT160 boards and eight Dialogic® DM/V4800BC boards. In order to resolve this issue, FSK and TrueSpeech features were removed from Media Load 2 and are no longer available. Refer to the Media Load 2 heading in Section 1.6, "New Media Loads for Dialogic® DM/V4800BC Media Boards Using Special Coders".
Resolved	IPY00099876	--	201	CSP	No termination reason is received from ATEC_TERMMSK( ).
Resolved	IPY00033059	--	111	CSP	When running CSP tests, some stuck channels are seen.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00045524	--	189	Device Management	The **dev_connect( )** function fails when used between M3G and DNI devices.
Resolved	IPY00091941	--	201	Diagnostics	A power management entry is marked as an ERROR in the system event log.
Resolved	IPY00045456	--	178	Diagnostics	The ISDNtrace tool failed to work for Dialogic® DM3 Boards configured for DASS2. **Note:** Contrary to the way ISDNtrace displays the data for Q.931 protocols, DASS2 link data is not translated and is provided raw, displayed as hex values instead.
Resolved	IPY00045159	--	178	Diagnostics	When using the PSTN Diagnostics tool (pstndiag), the following error occurs after clicking on a channel of an installed Dialogic® DM3 PSTN Board: “Error: Can't read alarms Trans (0x1616): no such element in array.”
Resolved	IPY00037708	--	135	Diagnostics	The its_sysinfo tool, which is used to collect data including a firmware dump, is not collecting a full memory dump.
Resolved	IPY00041078	--	152	DM/IP Boards	Unknown audio or DTMF is being sent from a Dialogic® DM/IP Board at the beginning of a SIP call, which precedes the expected audio to be heard from the file played.
Resolved	IPY00038391	--	148	DM/IP Boards	Dialogic® DM/IP Board stops returning events due to a DSP failure.
Resolved	IPY00038190	--	143	DM/IP Boards	When running high volume load tests with Dialogic® DM/IP601 cPCI boards to test SIP call control and media activity, it appears that firmware component is experiencing intermittent data access exceptions from “Task:0x1993418 StatesTask.” During this time, all active calls get suspended, and performing media activity is not transmitted across the network to other end-point.
Resolved	IPY00033472	--	120	DM/IP Boards	Specifying multiple DTMF detection methods prevents fax CED detection.
Resolved	--	--	145	DM/IP Boards	IP address of “0.0.0.0” is returned when calling **ipm_GetLocalMediaInfo( )** to obtain RTP port address after hot swap is done on Dialogic® DM/IP Board.
Resolved	IPY00079160	--	185	DM3 Call Control	When using NI2 protocol on Dialogic® DM3 Boards, **gc_GetCallInfo( )** did not retrieve the ANI when Numbering Plan ID was “Private”; it returned with a blank ANI.
Resolved	IPY00045132	--	175	DM3 Call Control	Under heavy load on certain Dialogic® DM3 PSTN Boards, calls that are offered to channels might get dropped immediately. They moved from Answering to Disconnected state due to an overlapping of call indexes among multiple network interfaces.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00044730	--	176	DM3 Call Control	When using Dialogic® DM3 Boards and ISDN protocol, the incorrect number of called subaddress digits is received by the application. An 8-digit called subaddress was initially sent by the switch, but the application received a 6-digit called subaddress.
Resolved	IPY00041407	--	157	DM3 Call Control	When setting up NFAS for the 4 lines on a Dialogic® DM3 T1 Board using DMS protocol. the board does not respond to a RESTART ACKNOWLEDGEMENT transmitted.
Resolved	IPY00041233	--	152	DM3 Call Control	When a call is terminated in the GCST_DETECTED state, a fake GCEV_OFFERED event should not be generated if the application has enabled the GCEV_DETECTED event.
Resolved	IPY00041209	--	152	DM3 Call Control	GCEV_UNBLOCKED event doesn't arrive for individual channels, even though GCEV_BLOCKED was delivered to individual channels, after AIS alarms occur and are then cleared.
Resolved	IPY00037507	--	135	DM3 Call Control	Event API fails to deliver an event when the T1 is configured for CAS and the cable is unplugged.
Resolved	IPY00034857	--	135	DM3 Call Control	When performing call progress analysis via the Global Call mediadetected method, if the media detection occurs before the out-of-band CONNECT message is received, GCCT_UNKNOWN is returned as a result.
Resolved	IPY00045184	--	176	DM3 Conferencing	The **dcb_dsprescount( )** function returned the incorrect number of resources for Dialogic® DM/IP241-1T1 Boards.
Resolved	IPY00044132	--	174	DM3 Conferencing	The **dcb_unmonconf( )** function removes the wrong conferee (party) from the conference. Instead of removing the monitor conferee, the most recently added conferee is removed from the conference.
Resolved	IPY00037861	--	135	DM3 Conferencing	If one conferee goes on mute, other conference participants hear buzzing noise. **Note:** A documentation update to Section 6.8, [0x3b] Parameters (parameters 0x3b03 and 0x3b04) has been added in the Documentation Updates section for the Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide. Please refer to it for information relevant to this defect resolution.
Resolved	IPY00037817	--	135	DM3 Conferencing	When playing background music through the telephone set, music cuts are heard when party A speaks.
Resolved	IPY00037396	--	135	DM3 Conferencing	Static background noise trails voice in conferences with more than 6 parties.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00037373	--	135	DM3 Conferencing	In a conference with two parties, if party A keeps speaking while party B starts speaking, party B hears breaks from party A while party B is speaking.
Resolved	IPY00093701	--	199	DM3 CSP	After running for approximately four hours, CSP devices get in a stuck state and the **ec_stream( )** function fails to start.
Resolved	IPY00045442	--	176	DM3 CSP	Dialogic® DM3 Board channel hangs when failing to listen to a TDM bus time slot prior to invoking the **ec_stream( )** function.
Resolved	IPY00045376	--	176	DM3 DTI	After setting event masks and then trying to retrieve the masks using **dt_getevtmsk( )**, this function failed with “Unknown error” as the reason.
Resolved	IPY00092493	--	199	DM3 Fax	Received fax pages contain either an incorrect "TO:" ID or the "TO:" header line is left blank.
Resolved	IPY00043443	--	165	DM3 Fax	Fax TIFF files are received with incorrect width; half pages are received.
Resolved	IPY00043240	--	165	DM3 Fax	An exception is generated during fax call tear-down process after **fx_stopch( )** is issued, causing the application to stop running.
Resolved	IPY00042934	--	162	DM3 Fax	Application exception occurs when calling **fx_setuio( )** and causes application to stop running.
Resolved	IPY00041421	--	154	DM3 Fax	Fax channels may hang when a stop is issued at the end of a send fax page.
Resolved	IPY00041079	--	154	DM3 Fax	The **fx_rcvfax( )** function returns -1 error after the system is running for several days, and the system is not able to receive faxes.
Resolved	IPY00039661	--	152	DM3 Fax	**ATFX_RESLN( )** sometimes returns 0, which is an invalid value. (According to the documentation, the only valid values are 98 and 196.) **Note:** A documentation update has been added in the Documentation Updates section for the Dialogic® Fax Software Reference. Please refer to it for information relevant to this defect resolution. There are additional return values for **ATFX_RESLN( )**, and the values passed to **fx_rcvfax( )** and **fx_sendfax( )** have more options. (The defect number associated with the documentation update is IPY00040796.)
Resolved	IPY00039476	--	148	DM3 Fax	Stuck fax channels during inbound calls.
Resolved	IPY00038407	--	139	DM3 Fax	**ATFX_RESLN( )** sometimes returns 0, which is an invalid value. (According to the documentation, the only valid values are 98 and 196.)
Resolved	IPY00037166	--	135	DM3 Fax	After an inbound fax call, the fax resource cannot go back to idle after **fx_stopch( )**.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00032797	--	135	DM3 Fax	The fax sender cannot wait to receive retry of digital identification signal (DIS) message, and gets Phase E status (EFX_COMMERRTX) transmit communication error.
Resolved	IPY00078445	--	178	DM3 Firmware	When using Dialogic® DM3 DM/V-B Boards, DSP crashes “KillTask” were seen in the DebugAngel log during standard playback; the affected channels could not be recovered without a board reinitialization.
Resolved	IPY00045440	--	176	DM3 Firmware	Dialogic® DM/V-B Board could not detect DTMF digits when digits are on about 0 dB per frequency.
Resolved	IPY00045388	--	176	DM3 Firmware	Playing a wave file with an invalid byte count in the header caused Dialogic® DM3 firmware to crash.
Resolved	IPY00045277	--	176	DM3 Firmware	An intermittent, partial PCM data stream corruption on network interface channels on Dialogic® DM/V1200BTEP Media Boards was observed during a local loopback mode test where every channel loops back its incoming stream out to the network.
Resolved	IPY00044832	--	174	DM3 Firmware	Under certain conditions and tone templates for a **dx_playtone( )** in asynchronous mode, the application does not receive the TDX_PLAYTONE event on the voice channel.
Resolved	IPY00043077	--	168	DM3 Firmware	Inbound ISDN calls made from cell phone failed to get answered on system connected to NET5 line. The calls get rejected with STATUS message with cause value of “100”, which indicates invalid information element contents. The Progress Indicator IE containing Progress Description 16 and Progress Location 1 is being rejected.
Resolved	IPY00041740	--	157	DM3 Firmware	Local pool size for GTD needs to be increased for Dialogic® DMV-B Boards.
Resolved	IPY00041580	--	157	DM3 Firmware	Over time (usually past 24 hours), dialed DTMFs get corrupted. When a '5' is dialed, the digit might get repeated in fragmented stutter and/or followed by an '8' even though never specified in the dialstring.
Resolved	IPY00043545	--	166	DM3 Host Runtime Library	Call to **ipm_Close( )** hangs with Dialogic® DI/0408-LS Switching Board if there are no additional DM3 voice resources in the system.
Resolved	IPY00043430	--	165	DM3 Host Runtime Library	After performing a peripheral hot swap of Dialogic® DM/IP Boards, and boards are successfully restarted, **gc_OpenEx( )** fails to open on-board IPML devices, returning error 0x44, Invalid Parameter.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00038998	--	143	DM3 Host Runtime Library	Bipolar violation alarms are properly detected by the board but are not being reported programmatically via GCAMS.
Resolved	IPY00092923	--	199	DM3 Network	The LineAdmin Utility does not view all information for DNI boards with more than four network interfaces.
Resolved	IPY00038533	--	143	DM3 Runtime Libraries	An internal parameter is not decremented correctly when a process exits, causing failures in opening devices.
Resolved	IPY00093815	--	199	DM3 Voice	A Blue Screen of Death (BSOD) occurs during heavy memory usage with no notification to the application.
Resolved	IPY00091452	--	195	DM3 Voice	A crash occurs in the Dialogic Cheetah library.
Resolved	IPY00082087	--	201	DM3 Voice	A Blue Screen of Death (BSOD) occurs when issuing an asynchronous call and exiting a thread. Fault points to dlgcmpd.sys.
Resolved	IPY00081776	--	201	DM3 Voice	An exception occurs when using the **dx_reciottdata( )** function with CreateFile( ) for opening the file and UIO on a DTMF terminated recording.
Resolved	IPY00079212	--	186	DM3 Voice	After **dx_getdig( )** was called and returned, and **dx_clrdigbuf( )** was called and returned, **ATDX_BUFDIGS( )** still reported that there were digits in the buffer; it should have returned 0. The problem occurred when **dx_getdig( )** terminated due to IDDTIME (inter digit delay).
Resolved	IPY00079095	--	185	DM3 Voice	Under certain race conditions, a **dx_playiottdata( )** caused an internal thread deadlock in the Voice library, leading to an application core dump.
Resolved	IPY00045293	--	176	DM3 Voice	Dialogic® DM3 Board channel hangs when failing to listen to a TDM bus time slot prior to invoking a record operation (**dx_reciottdata( )** or similar voice recording function).
Resolved	IPY00044932	--	174	DM3 Voice	A voice stuck channel occurred during a playback operation while running an application that is media intensive (plays/records) due to an internal race condition. No TDX_PLAY event is returned to the application after **dx_playiottdata( )** is issued.
Resolved	IPY00044811	--	174	DM3 Voice	A voice stuck channel occurred during a playback operation while running an application that is media intensive (plays/records) due to an internal race condition. No TDX_PLAY event is returned to the application after **dx_playiottdata( )** is issued.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00044614	--	174	DM3 Voice	A potential voice stuck channel condition that could occur on Dialogic® DM/V-B Boards during a record (**dx_reciottdata( )**) operation was caused by a race condition in the record data stream handling when a stop channel (**dx_stopch( )**) is issued to terminate the operation.
Resolved	IPY00044561	--	174	DM3 Voice	A voice stuck channel condition occurred, caused by the unhandling of tone creation failures during the setting of tone termination conditions for a playback (**dx_playiottdata( )**) or record (**dx_reciottdata( )**) operation. In asynchronous mode the TDX_PLAY or TDX_RECORD events, respectively, would never be received.
Resolved	IPY00044432	--	174	DM3 Voice	A firmware stream corruption was seen during the execution of a very intensive media application under specific conditions; this would cause a single voice channel not to return a TDX_RECORD event after **dx_reciottdata( )** function had been issued, causing the channel to be stuck.
Resolved	IPY00044363	--	174	DM3 Voice	A firmware DSP crash occurred during the running of an application that is media intensive (plays/records) due to an internal race condition; at some point plays start to fail simultaneously on multiple voice channels with RTF logs showing “Std_MsgError in PlayerStartingPlayer : Error Message 0x128” on groups of voice channels when attempting to execute a **dx_play( )** API call; similar results would occur on any other I/O voice function.
Resolved	IPY00044185	--	174	DM3 Voice	The **dx_setevtmsk( )** function with DM_DIGOFF only disabled events if DM_DIGITS was used first, i.e., they had to be first enabled; thus default user-defined tones could not be disabled.
Resolved	IPY00043907	--	174	DM3 Voice	A **dx_stopch( )** issued during a narrow window of time from a **dx_play( )** start caused the voice channel to get stuck in this state and never return a termination event TDX_PLAY.
Resolved	IPY00043818	--	168	DM3 Voice	Receipt of TDX_RECORD events is delayed after calling **dx_stopch( )** on Dialogic® DM/V4800BC Boards configured with the ML1B load at full density and under high load.
Resolved	IPY00043701	--	168	DM3 Voice	No TDX_PLAY event is ever returned from **dx_stopch( )** when the stop is issued during a playback operation and the time of the request coincides with some internal playback setup states timing window on the affected channel.
Resolved	IPY00043432	--	165	DM3 Voice	The **dx_playiottdata( )** function returns TDX_ERROR when attempting to play an empty wave file on Dialogic® DM3 Boards.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00042860	--	168	DM3 Voice	Receipt of TDX_RECORD events is delayed after calling **dx_stopch( )** on Dialogic® DM/V4800BC Boards configured with the ML1B load at full density and under high load.
Resolved	IPY00042845	--	162	DM3 Voice	Single channel play failures occur when running an application that handles a lot of plays/records concurrently and repeatedly in a live environment. At some point, a single play fails to return a TDX_PLAY event, and even calling **dx_stopch( )** will leave that channel in a stuck state.
Resolved	IPY00042828	--	162	DM3 Voice	Indexed wave file doesn't play as per the given indexes to **dx_playiottdata( )**; instead, the complete file is played.
Resolved	IPY00040832	--	152	DM3 Voice	TEC_STREAM event is not returned to the application when **ec_stopch( )** is called after **dx_unlisten( )** is performed on that voice channel.
Resolved	IPY00040685	--	152	DM3 Voice	**ATDX_TRCOUNT( )** returns the wrong value when playing a GSM 6.10 WAVE file on Dialogic® DM3 Boards.
Resolved	IPY00039586	--	145	DM3 Voice	ERROR_BROKEN_PIPE error internal message is reported in RTF logs during a streaming to board play.
Resolved	IPY00039412	--	145	DM3 Voice	TDX_PLAY is not generated to the application during streaming to board play; **dx_GetStreamInfo( )** is not returning correct information.
Resolved	IPY00039032	--	143	DM3 Voice	Voice resources don't go to idle state after **dx_stopch( )** function.
Resolved	IPY00038991	--	143	DM3 Voice	Previously existing user-defined tones are still being detected after deletion (i.e., call **dx_deltones( )**) on the same channel in which a new set of different user-defined tones have been created.
Resolved	IPY00038981	--	145	DM3 Voice	TDX_PLAY is not generated to the application during streaming to board play; **dx_GetStreamInfo( )** is not returning correct information.
Resolved	IPY00038611	--	139	DM3 Voice	When using the **dx_playtone( )** function with TONEON or TONEOFF as the terminating condition, when the TONEON or TONEOFF event occurs, the program gets a TDX_ERROR event instead of TDX_PLAYTONE event.
Resolved	IPY00037493	--	128	DM3 Voice	When running high volume load tests (500+ voice channels) for performing records, the RTF log shows "Buffer is corrupted" errors in the DM3 StreamSink component.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00037262	--	128	DM3 Voice	Under certain corner conditions, host CPU utilization increases a large percentage (15% witnessed) after issuing a record on multiple voice channels, and remains that way even after the record completes.
Resolved	IPY00037183	--	135	DM3 Voice	When recording WAV 176 bps file (11 KHz, 16 bits per sample), **dx_mreciottdata( )** stops prematurely with EOD before recording all bytes specified in io_length field of DX_IOTT structure, if this field is set to some large value (in this case, 26 Mb). Other formats, such as 64 kbs PCM MuLaw, ALaw, Linear, and ADPCM did not exhibit this problem.
Resolved	IPY00036865	--	135	DM3 Voice	If a user attempts to do a play forever (specifying io_length = -1) with UIO plays on DM3 Boards, there is still a hard upper limit on the number of bytes that can be played, which is approximately equal to 2.147 GB (~2 to the 31 bytes).
Resolved	IPY00036861	--	127	DM3 Voice	When attempting to run transaction recordings under rapid succession, sometimes the internal CT Bus routing fails and the record returns with a TDX_ERROR event with the result “Switching Handler is not Present.”
Resolved	IPY00028385	35433	--	DMN160TEC Boards	The registry specifies that the OffDiagPCDFileName for the Dialogic® DMN160TEC Board is thirdrock_xscale_diag.pcd. This file does not exist in the release build.
Resolved	IPY00099941	--	201	Drivers	A Virus Scanner conflict with the Dialogic device driver caused a Blue Screen of Death (BSOD).
Resolved	IPY00099886	--	201	Drivers	A hardware conflict with the Dialogic System Release drivers causes a Blue Screen of Death (BSOD.)
Resolved	IPY00098919	--	201	Drivers	An access violation crash occurs in the libsrlmt.dll library.
Resolved	IPY00099630	--	201	Fax	The **fx_stopch**(EV_SYNC) function returns before the fax channel goes into the IDLE state.
Resolved	IPY00099099	--	201	Fax	The fax component crashes while receiving a digital command signal (DCS) frame resulting in the application being unable to receive a fax until the Dialogic service is restarted.
Resolved	IPY00094517	--	201	Fax	The message confirmation (MCF) fax tone is incorrectly detected by DM3 as a partial page sent (PPS) tone.
Resolved	IPY00094385	--	199	Fax	An access violation in the internal library causes a race condition.
Resolved	IPY00092912	--	198	Fax	Fax calls cannot be made or received due to an access violation in the internal library.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00080931		201	Fax	A memory access violation error occurs when using fax resources and running an application in debug mode.
Resolved	IPY00100303	--	201	Firmware	The ml2e_cpciresb and ml5e_cpciresb media loads experienced hung channels.
Resolved	IPY00094251	--	199	Firmware	The encoder or decoder list is not populated after the board is downloaded.
Resolved	IPY00091410	--	195	Firmware	A firmware crash occurs when using the Dialogic® DM/V4800BC media load, ml1b.
Resolved	IPY00034036	--	122	Gatekeeper Registration	The gatekeeper may not respond to keep-alive registration within 2 seconds, causing GCEV_TASKFAIL at application level.
Resolved	IPY00092990	--	198	Global Call	Channels periodically hang while waiting on events from the **gc_Listen( )** function.
Resolved	IPY00033499	--	111	Global Call	The opening of dti devices via Global Call SS7 library fails.
Resolved	IPY00009930	35528	--	Global Call	GCEV_D_CHAN_STATUS not received after **gc_SetConfigData( )** is issued successfully to switch the variants on ISDN.
Resolved	IPY00093270	--	199	Global Call IP (SIP)	A memory leak is observed during a SIP ReINVITE.
Resolved	IPY00092647	--	198	Global Call IP (SIP)	The gc_MakeCall( ) function fails due to a glare condition.
Resolved	IPY00081061	--	192	Global Call IP (SIP)	The SIP stack does not negotiate correctly when VAD is specified as DON'TCARE on the local side. VAD-specific coders in an incoming SDP are not accepted.
Resolved	IPY00080944	--	192	Global Call IP (SIP)	Inbound calls are rejected due to maximum call legs allocated even though there is sufficient capacity on the node.
Resolved	IPY00080772	--	192	Global Call IP (SIP)	183 messages cannot be PRACKed if they generate a GCEV_PROGRESSING event.
Resolved	IPY00080516	--	191	Global Call IP (SIP)	After calling the **gc_AnswerCall( )** function, the application receives a GCEV_TASKFAIL event and IPT channels are lost.
Resolved	IPY00080340	--	191	Global Call IP (SIP)	The application now has the ability to add/append the Reason headers on outgoing BYE and CANCEL requests.
Resolved	IPY00080011	--	193	Global Call IP (SIP)	A GCEV_TASKFAIL event is returned on receipt of a LowBitRate HOLD reINVITE 200_OK.
Resolved	IPY00079716	--	189	Global Call IP (SIP)	Incoming calls are rejected with a GCEV_TASKFAIL (IPERR_INVALID_PHONE_NUMBER) event.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00079691	--	189	Global Call IP (SIP)	IPT responds to a reINVITE but loses RFC2833 indication on 200_OK.
Resolved	IPY00079668	--	189	Global Call IP (SIP)	IPT G729AB reINVITE reports IP_CAPABILITY as G729A.
Resolved	IPY00079651	--	189	Global Call IP (SIP)	The **gc_AcceptModifyCall( )** function fails with IPERR_BAD_PARAM.
Resolved	IPY00079648	--	189	Global Call IP (SIP)	The IPT reINVITE reports pTime in the IP_CAPABILITY structure.
Resolved	IPY00079108	--	189	Global Call IP (SIP)	IPPARM_OFFERED_FASTSTART_CODER/ GCSET_CHAN_CAPABILITY reporting for SIP G723.1 is always chosen at the default bit rate of 6.3k if remote side does not specify the bit rate.
Resolved	IPY00042168	--	158	Global Call Protocols	When running United States T1 Bidirectional protocol (pdk_us_mf_io), blind transfer failure scenario is not handled properly. In a failing scenario where **gc_BlindTransfer( )** yields a GCEV_DISCONNECTED event to the application, a subsequent **gc_DropCall( )** produced no GCEV_DROPCALL event as it should have.
Resolved	IPY00028383	35321	--	Global Call Protocols	Busy tones are detected as “no ringback” in call progress analysis when using the **dx_dial( )** method in Global Call application.
Resolved	IPY00028378	34586	--	Global Call Protocols	For inbound call, channel is blocked after the remote caller hangs up before sending DNIS, when using pdk_hk_dtmf_io.cdp.
Resolved	IPY00010746	35042	--	Global Call Protocols	When using the pdk_us_mf_io protocol, if CDP_OUT_Send_Alerting_After_Dialing = 1 and CPA is disabled, the user expects to get the GCEV_ALERTING event right after dialing. However, if the remote side answers the call too quickly, the GCEV_CONNECTED event is returned and the GCEV_ALERTING event never comes in.
Resolved	IPY00010621	34537	--	Global Call Protocols	When using the pdk_us_mf_io protocol in the Feature Group D configuration, ANI is missing the last digit when ANI is not terminated with the expected ST digit.
Resolved	IPY00010372	35035	--	Global Call Protocols	After sending CAS_HOOKFLASH, there should be some delay before sending DTMF in pdk_sw_e1_necls_io protocol, if CDP_WaitDialToneEnabled = 0 (i.e., do not wait for dialtone).
Resolved	IPY00010223	34985	--	Global Call Protocols	pdk_sw_e1_ermx_io.cdp can only accept one ringing signal (the internal ringing or the external ringing but not both). Defining CAS_RING_APPLIED (0001 -> 0xxx) solves the detection of the two ringing signals but causes problems with outgoing calls.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00010129	34274	--	Global Call Protocols	Global Call does not provide a way to disable DISCONNECT TONE SUPERVISION with pdk_na_an_io.cdp.
Resolved	IPY00010035	35159	--	Global Call Protocols	Under certain conditions when a **gc_MakeCall( )** attempt times out, it incorrectly displays the result message as NORMAL CLEARING instead of timeout.
Resolved	IPY00010004	34685	--	Global Call Protocols	When using the pdk_us_mf_io protocol in the Feature Group D configuration, the protocol does not send a Disconnect signal when it times out waiting for DNIS and ANI. This occurs when the remote side is configured as Feature Group B and makes a call.
Resolved	IPY00009837	35049	--	Global Call Protocols	There seems to be a hard-coded 30-second timeout on a Make Call when the call is made in Alerting mode, which will terminate the call if no one picks up the phone. The expected behavior is that the call will not be dropped automatically, so the phone will ring forever if no one picks up. This occurs on T1 CAS lines.
Resolved	IPY00009409	34663	--	Global Call Protocols	When using FXS protocol and calling a busy station using supervised transfer, you get a disconnect event for both the consultation CRN and transferred CRN.
Resolved	IPY00008220	34972	--	Global Call Protocols	When using the pdk_us_mf_io protocol, digits from the previous call are returned in ANI.
Resolved	IPY00007327	30233	--	Global Call Protocols	With the pdk_mx_r2_io protocol, if the E1 cable is disconnected and reconnected, the application does not receive all the GCEV_UNBLOCKED events.
Resolved	IPY00006809	34543	--	Global Call Protocols	When CDP_IN_DNIS_ST_Needed = 0, the pdk_e1_cas_io protocol should not issue timed-out error while waiting for DNIS.
Resolved	IPY00006804	34319	--	Global Call Protocols	If a board is configured using pdk_us_ls_fxs_io.cdp file and a call is abandoned after the first ring, the application is not receiving the GCEV_DISCONNECTED event that is expected.
Resolved	IPY00006771	34329	--	Global Call Protocols	Using Belgium R2 protocol, when configured in DTMF/MF mode, in the OFFERED state the **gc_ResetLineDev( )** function does not behave properly.
Resolved	IPY00006762	34664	--	Global Call Protocols	When using E1 line side protocol and calling a busy station using supervised transfer, you get a disconnect event for both the consultation CRN and transferred CRN.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00006748	34587	--	Global Call Protocols	The PDK E1 CAS protocol cannot be downloaded on Dialogic® DM3 Boards, and Dialogic® Springware Board channels cannot be opened when using this protocol.
Resolved	IPY00006735	34344	--	Global Call Protocols	On DM3, when dialtone is enabled on Belgium R2 protocol, if the first DTMF/MF digit of DNIS sent is 1 then the DNIS digits received at the inbound side are not the same as sent by the outbound side.
Resolved	IPY00037372	--	135	H.323 Call Control	An access violation/assert is seen in the Global Call IP Call Control library if a RequestMode message for changing audio codecs is received.
Resolved	IPY00037351	--	135	H.323 Call Control	When the remote capabilities contain one audio codec and T.38 fax codec, the Global Call IP Call Control Library will incorrectly attempt to switch to fax.
Resolved	IPY00036799	--	127	H.323 Call Control	The calling party number (CGPN) and called party number (CDPN) IEs can be queried but cannot be set in H.323.
Resolved	IPY00033058	--	108	Host Admin	In log file, the OSSL throws an exception on thread WaitForCompletion. The OAMTrainsport call does not catch the exception on every OSSL call.
Resolved	IPY00010194	35056	--	Host Admin	The **dx_getcachesize( )** returns cache availability even though it is not sufficient enough to play prompts.
Resolved	--	--	176	Host Install	A problem has been observed on multiple Windows® 2003 systems where an error occurs during the installation of the Dialogic® System Release Software. A pop-up error message box is displayed when the installation of the DetectorsProj service fails. It has been determined that this error is due to some other software package de-registering the Microsoft® *ATL.DLL* file. This file is not delivered as part of the Dialogic® System Release Software. If you observe this failure, the following steps can be used to resolve the problem: 1. Change into the Windows System32 directory. 2. Run “regsvr32 atl.dll”. 3. Change into the Dialogic\bin directory. 4. Run “DetectorsProj -service”. This can be done once the Dialogic® System Release Software installation has completed with the above error, before rebooting. Since this service is not started automatically, the above commands can also be done after reboot, but before the Dialogic® boards are started. It is only necessary to perform this procedure one time.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00035831	--	122	Host Library	Segmentation fault occurs in libipm_vsc.so when calling **gc_close( )** on Global Call (IP based) line device.
Resolved	IPY00038849	--	143	Host Runtime Libraries	When opening channels asynchronously with **gc_open( )**, sequentially one after another channels fail to open.
Resolved	IPY00009315	35557	--	Host Runtime Libraries	Board download throws PDKManager error when two or more variants of the CAS protocol are selected in the *pdk.cfg* file.
Resolved	IPY00093410	--	201	Installation	The upgrade install option deletes the pdk.cfg file.
Resolved	IPY00092250	--	201	Installation	INVALID HANDLE prints are repeatedly written to the *dlgagent.log* file.
Resolved	IPY00092204	--	201	Installation	The software uninstall utility hangs for up to 30 minutes as it removes the /dialogic folder.
Resolved	IPY00033563	--	122	IP	SIP to SIP speech path broken between 180 Ringing(SDP) and 200 OK.
Resolved	IPY00092212	--	196	IP Host	An 18x (0.0.0.0) response to multiple codec offers causes incorrect handling of the INVITE.
Resolved	IPY00092028	--	196	IP Host	The remote payload type received in SDP causes the application to crash.
Resolved	IPY00044700	--	174	IP Host	An incorrect processing of certain H.323 parameters in a call disconnect caused a library exception condition that trickled up to the customer application as a crash.
Resolved	IPY00044544	--	174	IP Host	Placing a SIP call that sends an INVITE message with certain length SIP diversion header field contents results in a core dump. This only occurs when the gc_h3r RTF logging module is enabled.
Resolved	IPY00044215	--	174	IP Host	A Dr. Watson event occurs, which takes the application down. Stack traces indicate that there is an exception firing in strcpy in one of the Dialogic threads.
Resolved	IPY00043307	--	170	IP Host	After performing a peripheral hot swap of Dialogic® DM/IP Boards, and boards are successfully restarted, **gc_OpenEx( )** fails to open on-board IPML devices, returning error 0x44, Invalid Parameter.
Resolved	IPY00042528	--	160	IP Host	An unhandled error case in an IP Host library allowed the processing of certain bad messages that lead to a process crash under certain SIP application on Dialogic® IPT Boards. The application log in Windows® event viewer shows dlgsysmonitorserver.exe - Application Error...The memory could not be read; the fault module was pmac_transport.dll.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00037004	--	127	IP Host	IP channels sporadically fail to receive Global Call events resulting from **gc_AnswerCall( )**, **gc_DropCall( )**, or **gc_Extension( )** calls.
Resolved	IPY00033102	--	120	IP Host	Supervised transfer fails. Party B gets GCEV_XFER_FAIL with “reason 5800”.
Resolved	IPY00044273	--	174	IP Media Session Control	A request to re-INVITE from audio to T.38 fax using **gc_ReqModifyCall( )** in IP_T38_MANUAL_MODIFY_MODE fails with GCEV_MODIFY_CALL_FAIL event.
Resolved	IPY00081853	--	194	IP Media Session Control RTP	The Dialogic® IPT Board sporadically sends RTP with the destination MAC address set to all zeros (for example, 00.00.00.00.00.00).
Resolved	IPY00092115	--	196	IPT Boards	Resetting IPT device media addresses causes a loss of speech path.
Resolved	IPY00081301	--	193	IPT Boards	An internal IPT device hardware malfunction prevented completion of the SIP or H.323 call control message on the channel. As a result,all subsequent outbound call attempts failed with an "Invalid State" error message. See the Dialogic® Global Call IP Technology Guide document update regarding the new error message related to this issue.
Resolved	IPY00079365	--	186	IPT Boards	A Dialogic® IPT Board channel locked when it failed to connect to the remote media address. The problem occurred when the application made an outbound IP call from the IPT cPCI host, and the remote carrier responded with an 18x (SDP) containing their remote media IP information. If that remote media IP address is unreachable, the IPT Board does not post either the GCEV_EXTENSION events with the media connection details, nor does it post a subsequent (and required) GCEV_CONNECTED. The channel becomes unusable, and a reboot is required.
Resolved	IPY00078978	--	185	IPT Boards	A Dialogic® IPT Board channel locked when it failed to connect to a remote media address. The channel became unusable and required a board reinitialization.
Resolved	IPY00045304	--	176	IPT Boards	On Dialogic® IPT Boards, **ipm_StartMedia( )** or **ipm_ModifyMedia( )** caused 5 to 20 msec of invalid data injected toward the CT Bus. This was observed with DefaultTimeslotEncoding and CT Bus settings in either companding setting (A- or U-law), with G.711 IP sessions. This invalid data caused audible clicks on the TDM side of the session.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00044779	--	174	IPT Boards	**gc_OpenEx( )** fails with error message "IPERR_INTERNAL" following a successful single board stop, removal, re-insertion, and single restart of a Dialogic® IPT Board; meanwhile, a second board was actively processing calls throughout the hot swap process on the former.
Resolved	IPY00044699	--	174	IPT Boards	IPMEV_ERROR events are returned to the application resulting in RTP audio streaming failures, when calling **ipm_StartMedia( )** on multiple channels during a bulk load test on Dialogic® IPT4800C Board.
Resolved	IPY00044199	--	170	IPT Boards	After performing a peripheral hot swap of Dialogic® IPT Boards, and boards are successfully restarted, **gc_OpenEx( )** does not return a result and causes the application to hang.
Resolved	IPY00043801	--	170	IPT Boards	The PMAC transport RTF logging module generates multiple "error 995" messages during startup of the Dialogic® Configuration Manager (DCM) when Dialogic® IPT Boards are in the system.
Resolved	IPY00043292	--	166	IPT Boards	SIP hairpin call fails with no audio sent from Dialogic® IPT4800C IP Board. The IPT4800C does not continue the transmission of RTP and RTCP packets when ICMP is received from remote SIP client.
Resolved	IPY00043267	--	166	IPT Boards	Audible clicking is heard in the audio stream on a system using Dialogic® IPT4800C IP Boards when the IPM resource was not listening to a CT Bus time slot; the silence pattern was not set properly according to the time slot encoding, instead the default PCM silence pattern 0x00 was transmitted.
Resolved	IPY00042866	--	166	IPT Boards	System crashes were experienced when using Dialogic® IPT4800C IP Boards along with a Dual Core SBC. The following error was seen: DLGC_EVT_CP_FAILURE(0x30001).
Resolved	IPY00042464	--	160	IPT Boards	Jitter alarms are repeatedly set ON and persist after the call is ended.
Resolved	IPY00042336	--	160	IPT Boards	RTP streaming from a Dialogic® IPT Board to a SIP client stops after approximately 3 seconds if the client does not send RTCP.
Resolved	IPY00042300	--	159	IPT Boards	Dialogic® IPT Board will not switch to T.38 if it rejected a previous audio re-INVITE.
Resolved	IPY00042204	--	159	IPT Boards	When using Dialogic® IPT Board, IP channel signaling APIs (**gc_MakeCall( )**, **gc_AnswerCall( )**, etc.) fail, and RTF logs indicate that local media information is invalid. Subsequent calls can only be completed successfully after a **gc_ResetLineDev( )** on that channel.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00041296	--	156	IPT Boards	Outgoing H.323 calls result in GCEV_TASKFAIL at the Alerting stage. After the call's release, no successful call can be made on that channel (iptB1Tx, ipmB1Cx device) until the board is reset.
Resolved	IPY00041280	--	156	IPT Boards	An exception is raised in the Dialogic® stack that leads to an application crash when attempting to make an outbound call if the remote end point offers fax-only codec on an H.323 slowstart call.
Resolved	IPY00040743	--	154	IPT Boards	In order to achieve a performance and debugging capabilities balance with Dialogic® IPT Boards, some logging considered excessive has now been eliminated.
Resolved	IPY00040179	--	148	IPT Boards	Cannot open a full (480 channel) Dialogic® IPT Board of 1PCC IP devices.
Resolved	IPY00039847	--	146	IPT Boards	Sending of a fax is unsuccessful when making a TDM fax call across two IPT machines joined by H.323 across an IP network. RequestMode is not triggered upon CED detection in AUTO mode.
Resolved	IPY00038060	--	140	IPT Boards	An assert occurs when there are no media attributes containing “rtpmap” in a SIP INVITE to a Dialogic® IPT Board.
Resolved	IPY00038365	--	140	IPT Call Control	Egress SIP calls work briefly, but then omit SDP in egress INVITE message.
Resolved	IPY00038240	--	140	IPT Call Control	An assert occurs on inbound re-INVITE.
Resolved	IPY00038551	--	143	Modular Station Interface (MSI)	**ms_stopfn( )** causes two Release Call messages to be sent to the DM3 firmware.
Resolved	IPY00038433	--	143	Modular Station Interface (MSI)	The **ms_stopfn( )** function fails to stop the ringing on Dialogic® DISI32R2 Board.
Resolved	IPY00044200	--	170	OA&M	The OAMSYSLOG module from RTF reports Dialogic® IPT Board errors when attempting to restore the board configuration from a previously saved one during system startup using the **NCM_RestoreConfiguration( )** API.
Resolved	IPY00042584	--	160	OA&M	Dialogic® DM3 Boards were showing incorrect PCI slot/bus numbers in the installed boards configuration section of the its_sysinfo tool, as compared to the initial assignments given by NCM/DCM.
Resolved	IPY00040536	--	152	OA&M	While application is running, message is logged in the Windows® event log: Faulting application OAMEventService.exe, version 1.0.0.21.
Resolved	IPY00038280	--	139	OA&M	A non-OAMIPC based client was attempting connection to an internal software component, an OAMIPC-based server. This caused the internal OAMIPC-based server to crash when invoking the Dialogic® system service startup or shutdown.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00078576	--	186	PDK	When the Brazil R2 Bidirectional protocol (pdk_br_r2_io) is configured for overlap send and the switch wants “silence” instead of “F” to represent end of DNIS, the ANI digits aren't sent after DNIS communication completes. When the first ANI digit was requested, no digits were sent. The protocol then timed out and the call failed. **Note:** A new parameter, **CDP_SKIP_A3_AND_A4_PULSE**, has been added to the *pdk_br_r2_io.cdp* file to handle this situation. For information about the new parameter, see the Documentation Updates section for the Dialogic® Global Call Country Dependent Parameters (CDP) for PDK Protocols Configuration Guide.
Resolved	IPY00042609	--	164	PDK Protocols	Examination of the CC Result information upon receiving a GCEV_TASKFAIL event with GCRV_PROTOCOL (0x503H) result value, showed undocumented error codes in the header file (*pdkerror_list.h).*
Resolved	IPY00041808	--	162	PDK Protocols	When running with any PDK protocol and the application is abruptly shut down, the existing calls are not dropped; furthermore, noise is heard at the terminating end due to in-band non-idle data being transmitted as a result of media exit sequence.
Resolved	IPY00041855	--	157	Protocols	Call could not be completed because the Mexico R2 protocol failed to send additional DNIS digits.
Resolved	IPY00079866	--	190	PSTN	No response is sent back to the application upon receipt of a user-to-user service 1 or 2 request.
Resolved	IPY00079825	--	190	PSTN	When receiving an IAM with the continuity check indicator set to spare (illegal value), the application handles it as if a "continuity check required" indicator was received.
Resolved	IPY00079551	--	189	PSTN	IAM messages exceed the maximum allowed.
Resolved	IPY00099200	--	201	PSTN Call Control	When a call is connected with Q.Sig protocol, the application is not able to send a facility message in an already connected NCAS call.
Resolved	IPY00099067	--	201	PSTN Call Control	Digital T1 ISDN channel lock up is caused by incorrect parsing of certain misconstrued High Layer Compatibility IEs received in the D-channel.
Resolved	IPY00093627	--	199	PSTN Call Control	Incorrect Diversion IE. Scenario: An outbound call is made and the switch informs that the number is diverted. The board needs to make the second call AND set the DIVERSION_IE to tell the new switch what number was originally dialed.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00092283	--	199	PSTN Call Control	The **gc_SetConfigData( )** function in asynchronous mode does not return the GCEV_SETCONFIGDATA event when setting CCSET_CALLANALYSIS setID on the GCTGT_CCLIB_CHAN target.
Resolved	IPY00092252	--	198	PSTN Call Control	Clocking issues and HDLC framing errors cause the system to stop responding.
Resolved	IPY00091862	--	199	PSTN Call Control	The application is unable to retrieve CLI from DPNSS SSRM(I).
Resolved	IPY00080244	--	190	PSTN Call Control	A memory leak occurs while receiving H.323 calls with a Global Call-based application.
Resolved	IPY00079477	--	188	PSTN Call Control	When using the NI2 protocol with NFAS configuration, the system couldn't make outbound calls. Also, when the switch sent a RESTART message to the Dialogic® Board, it returned a RESTART ACKNOWLEDGE with a different interface ID than the interface ID in the RESTART message.
Resolved	IPY00042601	--	164	PSTN Call Control	Segmentation fault happens when **gc_util_insert_parm_val( )** is called before **gc_Start( )**. Instead of a segmentation fault, the error should be reported gracefully.
Resolved	IPY00038979	--	143	PSTN Call Control	The pdk_sw_e1_fxs_io protocol does not forward the correct reason when a call is disconnected due to detection of a SIT. The reason should indicate that SIT was detected.
Resolved	IPY00038494	--	143	PSTN Call Control	CP failure on Dialogic® DM/N960-4T1 Board.
Resolved	IPY00038244	--	139	PSTN Call Control	If **gc_MakeCall( )** is called with GC_PARM_BLK set to NULL, ERR1 is shown in the RTF log.
Resolved	IPY00037841	--	139	PSTN Call Control	When using ISDN under DM3, **gc_open( )** fails after hot swap test. The dm3cclib cancels 2 events when closing devices (detected and offered), but only waits for one.
Resolved	IPY00037607	--	135	PSTN Call Control	If another call comes in between a **gc_DropCall( )** and **gc_ReleaseCallEx( )**, the call is not detected. The problem occurs when the drop call and release call are issued within 1-2 seconds of each other.
Resolved	IPY00036886	--	125	PSTN Call Control	The call type information is incorrectly being encapsulated in the METAEVENT's extevtdatap pointer in the GCEV_OFFERED event when using ISDN call control on Dialogic® DM3 Boards.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00036504	--	127	PSTN Call Control	Calling **gc_MakeCall( )** causes a SETUP message to be sent. If the first response from the other side is CONNECTED, the board responds with CONNECT_ACK, but GCEV_CONNECTED is not sent to the application. The problem only occurs if the board is set to Network End; if the board is set to User End, GCEV_CONNECTED is sent.
Resolved	IPY00036347	--	127	PSTN Call Control	QERROR_WARNING messages appear in *Dm3StdErr log*. Eventually, **gc_SetChanState( )** fails on all channels, and all channels are blocked.
Resolved	IPY00036337	--	125	PSTN Call Control	5ESS did not support CALLED NUMBER TYPE in the NETWORK_SPECIFIC (0x03) IE for DM3 Boards.
Resolved	IPY00036248	--	135	PSTN Call Control	When using Global Call SS7, the 0xb and 0xc address signals, which were previously reported to the application as “b” and “c”, are now getting reported as “#” and “*”, thus breaking backward compatibility.
Resolved	IPY00035451	--	120	PSTN Call Control	WinXP **gc_OpenEx( )** fails for device “:N_dkB1T1” for SS7 Board when configured for clear channel.
Resolved	IPY00035148	--	120	PSTN Call Control	The **gc_Unlisten( )** function has no effect when issued on “dk” devices using Global Call SS7.
Resolved	IPY00034618	--	120	PSTN Call Control	**gc_DropCall( )** fails when responding to a GCEV_DISCONNECT event after a GCEV_BLOCKED event.
Resolved	IPY00034606	--	123	PSTN Call Control	While issuing a make call during a supervised transfer to a destination that is busy, **gc_ResultMsg( )** function returns with PROTOCOL ERROR.
Resolved	IPY00034254	--	123	PSTN Call Control	**gc_SetConfigData( )** function hangs on changing line encoding/framing at runtime.
Resolved	IPY00033163	--	120	PSTN Call Control	Access violation occurs when running ISDN based application.
Resolved	IPY00039492	--	143	Runtime Trace Facility (RTF)	RTF logging has a memory leak and drops some log messages.
Resolved	IPY00038894	--	140	Runtime Trace Facility (RTF)	RTF logging corrupted device name in **dx_close( )**.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00038545	--	143	Runtime Trace Facility (RTF)	In RTFManager, the *RtfMatrix.xml* file was used to map the modules in the RTFConfig file to a family and technology group. But if any changes were made to the RTFConfig file outside of RTFManager, the configuration section would fail. **Note:** The mapping file was removed, and attribute tags were added to the RTFConfig file to define the mappings, making the configuration section of RTFManager more robust.
Resolved	IPY00038524	--	139	Runtime Trace Facility (RTF)	Multiple threads can be created in the RTF server for a single client when the system is heavily loaded. This leads to a build-up of threads in the server, which can lead to thread creation failures.
Resolved	IPY00036919	--	135	Runtime Trace Facility (RTF)	Unable to configure RTF trace capabilities using RTFManager because the selection is grayed out.
Resolved	IPY00036469	--	127	Runtime Trace Facility (RTF)	RTF 3.0 introduced increased memory usage of 7MB in the client. So for each process running on the system that is directly or indirectly linked with RTF, an additional 7MB of memory is used.
Resolved	IPY00093771	--	199	SIP Call Control	An access violation in the internal library causes the system to crash.
Resolved	IPY00093022	--	199	SIP Call Control	An incoming INVITE is sometimes rejected immediately with 603 DECLINE.
Resolved	IPY00079393	--	188	SIP Call Control	The SIP Allow header was missing in some responses to inbound calls.
Resolved	IPY00079251	--	185	SIP Call Control	The SIP Allow header does not include the “INFO” method. When acting as a SIP User Agent Server (UAS), 1xx and 200_OK messages do not include INFO as a listed method in the Allow header. When acting as a SIP User Agent Client (UAC), INVITE does not include the INFO method in the Allow header. **Note:** A documentation update has been added in the Documentation Updates section for the Dialogic® Global Call IP Technology Guide. The INFO method is now included as part of the Allow header in SIP messages by default.
Resolved	IPY00078519	--	178	SIP Call Control	The application sets a 5.3 Kbps bit rate for the G.723.1 transmit codec; however, the 6.3 Kbps rate for the codec is RTP transmitted instead, and reported as such by IPPARM_FASTSTART_CODER event.
Resolved	IPY00041300	--	154	SIP Call Control	SIP calls are rejected with a “486 Busy Here” due to incorrect handling of the scenario when a previous call was terminated due to bad incoming SDP.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00041118	--	154	SIP Call Control	The application is unable to make a SIP call using the **gc_MakeCall( )** function on the same channels previously used to make an H.323 call.
Resolved	IPY00039965	--	148	SIP Call Control	Outbound IP call fails with "IPEC_SIPReasonStatus503ServiceUnavailable" when the hostname is passed as the destination address in the dialstring. The outbound call using **gc_MakeCall( )** is not able to resolve the hostname to an IP address for the call to complete successfully.
Resolved	IPY00039707	--	148	SIP Call Control	Automatic SIP re-INVITE when media switches from audio to fax causes a glare condition that disconnects the call. **Note:** To resolve this issue, new Global Call parameters have been added to disable/enable the sending of the automatic SIP re-INVITE message upon media switch. For information about this feature, see Section 1.14, "Disabling Automatic re-INVITE Message when Switching between Fax and Audio", on page 51.
Resolved	IPY00039401	--	146	SIP Call Control	The "Record-Route" field of a SIP header message is incorrectly reported as the "Route" field when present within an incoming SIP message through use of the Dialogic® Global Call API.
Resolved	IPY00038572	--	140	SIP Call Control	When running a Dialogic® Global Call IP-based application that enables notification of SIP messages through GCEV_EXTENSION events, the type of SIP message received with the event cannot be identified. The message type value retrievable with that event returns more bytes than expected, making it unable to decipher which message was received.
Resolved	IPY00035822	--	122	SIP Call Control	Global Call SIP application does not respond to 407 Proxy Authentication Required messages.
Resolved	IPY00035806	--	122	SIP Call Control	Dialogic® IPT Board is sending "VADFLAG_B0: NON-speech" message in RTP stream even if VAD is disabled.
Resolved	IPY00035613	--	122	SIP Call Control	Fails to send a BYE message after dropping call on Avaya IP PBX.
Resolved	IPY00034765	--	120	SIP Call Control	MS RTC client cannot connect SIP calls with Dialogic® IPT6720 Boards using G.723 coder.
Resolved	IPY00033912	--	122	SIP Call Control	Early media on 180 Ringing(NO SDP) fails if T.38 required.
Resolved	IPY00033763	--	120	SIP Call Control	Dialogic® IPT Boards incorrectly timestamping packets.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00080020	--	190	SNMP	OID data type returned by the DLGHWINF, DLGSRPRF, DLGR4DEV, and DLGDS1 agents does not always match the type definitions in their respective MIB files.
Resolved	IPY00079590	--	189	SNMP	An incorrect Enterprise ID is returned
Resolved	IPY00045292	--	178	SNMP	When using SNMP and querying the dlgIsdnSigProtocol OID in the DLGCISDN MIB, it returns ''4ess''; the proper string returned should be ''4ess/Ni2” instead.
Resolved	IPY00036855	--	139	SNMP	When using MIB2 from RFC1213, Dialogic® SNMP agent fails to return valid information when a “get” command is issued.
Resolved	IPY00091022	--	198	SS7	SS7 software fails to start due to a tracing function defect.
Resolved	IPY00045239	--	176	SS7	The small window of time between the receipt of a GCEV_UNBLOCKED and **gc_WaitCall( )** completion was enough to miss GC/SS7 calls; the library discarded calls received during that window.
Resolved	IPY00045224	--	176	SS7	Dialogic® Global Call SS7 application did not work properly the first time after the Dialogic® SS7 Boards were initialized; the application needed to be torn down and brought up again for it to work properly.
Resolved	IPY00044425	--	175	SS7	A GCEV_OFFERED event was sent before the application issued **gc_WaitCall( )**; this is not the correct call flow order in Global Call.
Resolved	IPY00044100	--	175	SS7	The GC SS7 server log level configuration in *gcss7.cfg* is not working; whether the debug level is set to “All,” “None,” or “Errors,” the log file is the same as “All.”
Resolved	IPY00037918	--	139	SS7	The RSI link goes down intermittently.
Resolved	IPY00037767	--	135	SS7	The GCSS7 library does not generate the GCEV_MOREINFO event if it receives a SAM message with only STOP digit (0xf) after the application has already issued **gc_CallAck( )**.
Resolved	IPY00037632	--	135	SS7	If there is a delay in the SS7 server picking up messages from the IPC queue, an ERROR_IO_PENDING occurs and the SS7 library terminates the IPC. This causes all the circuits to get blocked, as there is no more connection with the SS7 service. This is causing the IVRs to get a sudden circuit block from the switch in all of its SS7 circuits.
Resolved	IPY00043230	--	166	Standard Runtime Library (SRL)	An access violation occurs during **sr_waitevtEx( )** processing.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description
Resolved	IPY00039334	--	154	Standard Runtime Library (SRL)	An application crash occurred; the stack trace shows SRL library at the top of the stack.
Resolved	IPY00039155	--	145	Standard Runtime Library (SRL)	An application crash occurs with SRL at the top of the stack; the SRL was not initializing all variables of a structure for a given thread, which can cause an access violation.
Resolved	IPY00038708	--	143	Standard Runtime Library (SRL)	An access violation occurs when application calls **sr_waitevtEx( )** for the same device on multiple threads.
Resolved	IPY00100083	--	201	Voice	A firmware crash leads to hung voice channels.
Resolved	IPY00099791	--	201	Voice	Voice channels become unusable after the dx_stopch( ) function fails to return a TDX_PLAY, and the dx_resetch( ) only returns TDX_RESETERR.
Resolved	IPY00094190	--	199	Voice	CSP channels hang after the system clock is moved ahead by 20 seconds or more.
Resolved	IPY00093957	--	201	Voice	A memory exception results when the application attempts to allocate 10 bytes for the DX_XPB data structure and pass it to the dx_playiottdata( ) function.
Resolved	IPY00080252	--	191	Voice	Voice Media intensive (plays/records) caused the play and record functions to fail. TDX_ERROR events with reason 0x80000 (system error) were observed.
Resolved	IPY00080145	--	190	Voice	The voice channel remains in a PLAYING state after the **dx_playiottdata( )** function returns a failure (-1) when called asynchronous mode.
Resolved	IPY00080009	--	190	Voice	An Access Violation is observed on the system.
Resolved	IPY00079561	--	190	Voice	ATDX_CRTNID returns a 0 instead of the proper value.
Resolved	IPY00079523	--	189	Voice	While retrieving board status, d42_getbrdstatus crashes in debug mode and returns an incorrect value in release mode.
Resolved	IPY00079353	--	189	Voice	Audio is missing at the end of recorded files.
Resolved	IPY00038435	--	139	Voice	Channels hang and are not able to recover once in a CS_STOPD state.
Resolved	IPY00037818	--	135	Voice API	The **dx_setevtmsk( )** function fails to disable the TDX_CST events for DE_DIGITS when setting the DM_DIGOFF flag. **Note:** A documentation update has been added in the Documentation Updates section for the Dialogic® Voice API Library Reference. Please refer to it for important information relevant to this defect resolution.

Issues Sorted By Type, Dialogic® System Release 6.1 CompactPCI for Windows® (Continued)

Issue Type	Defect No.	PTR No.	SU No.	Dialogic® Product(s) or Component(s)	Description

Resolved	IPY00037796	--	135	Voice API	TDX_RESETERR and TFX_FAXRECV events have the same value defined in their respective header files (*dxxxlib.h* and *faxlib.h*). This can lead to a conflict at the application level when performing either the voice or fax channel recovery feature; you cannot tell the difference between a pair of potential completion events when executing **dx_resetch( )** and **fx_rcvfax( )** API calls.
					Note: The fix for defect IPY00037796 changed the values of the two defines, TDX_RESET and TDX_RESETERR, in *dxxxlib.h* from 0xA1/A2 to the following:
					`#define TDX_RESET (DXXEV_BASE | 0x01) // reset event for Reset API's`
					`#define TDX_RESETERR (DXXEV_BASE | 0x02) // reset error event for Reset API's`
					Applications with DM3 Boards that use the **dx_resetch( )** function must be recompiled to work properly.

Resolved	IPY00037432	--	135	Voice API	The **dx_clrdigbuf( )** function overwrites area of thread's stack space, causing the application to crash.
Resolved	IPY00091543	--	195	Configuration	A memory leak occurs when using the Dialogic® NCM API functions.
Resolved	IPY00090734	--	195	DM3 Fax	A DM3 fax firmware operation fails and causes a KILLTASK.
Resolved	IPY00091039	--	195	Global Call IP (SIP)	Ingress/Egress calls are rejected because no call objects are available.
Resolved	IPY00091490	--	195	IPT Boards	Resetting IPT device media addresses causes loss of speech path.
Resolved	IPY00091224	--	195	ISDN	A Status message is transmitted following the receipt of ALERTING with two Progress Indicators.
Resolved	IPY00091379	--	195	Global Call IP (SIP)	A crash occurs in the LIBSIPSIGAL.DLL library.

Documentation Updates

This chapter contains information on updates and corrections to the documents included in Dialogic® System Release 6.1 CompactPCI for Windows®. Documentation updates are divided into the following categories:

3.1 System Release Documentation Updates

This section contains updates to the following documents (click the title to jump to the corresponding section):

- Dialogic® System Release 6.1 CompactPCI for Windows® Release Guide

3.1.1 Dialogic® System Release 6.1 CompactPCI for Windows® Release Guide

Updates to **Section 2.2, Basic Software Requirements**

Windows® Server 2003 SP2 should be listed as a supported operating system for the Service Update.

The following requirement should be added:

- Java Runtime Environment (JRE) version 1.5 must be installed on your system in order to run the diagnostic tools included with System Release 6.1 CompactPCI Windows.

In **Section 3.1, New DMV1200BTEC, DMV600BTEC and DMV4800BC Products**, refer to the Initial Alarm State (0x1626) description in the CONFIG File Parameter Reference chapter of the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* for more information about the following feature:

Trunk preconditioning

The ability to send an alarm state to the network at all times from powerup to application startup. Refer to the *Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide* for more information.

Update to Section 4.1, Installation (IPY00081147)

Add the following note after the first paragraph:

Note: Due to changes in the Dialogic® Software install process, an update install should not be used when updating to a more recent build from a Dialogic® System Release 6.1 CompactPCI for Windows build prior to Service Update 176. Update install does work when upgrading from Service Update 176 to any more recent build.

In **Chapter 6, New Global Call Features for ISDN Technology**, the "Programming Libraries" section describes the following feature but omits some of the boards supported:

Support for QSIG NCAS

The ability to make (outbound) and receive (inbound) Non-Call Associated Signaling (NCAS) calls is supported for the QSIG protocol (E1 or T1) on the Dialogic® DMV1200BTEC, DM/V960A-4T1-cPCI and DM/V1200A-4E1-cPCI Boards. The feature is only supported on media loads that use the QSIG T1 or E1 protocol; for example, ml2_qs2_qsige1. See the *Dialogic® Global Call ISDN Technology Guide* for more information. Note that the QSIG NCAS feature is also supported on the Dialogic® DM/IP products too.

3.2 Installation and Configuration Documentation Updates

This section contains updates to the following documents (click the title to jump to the corresponding section):

- Dialogic® System Release 6.1 CompactPCI for Windows® Software Installation Guide
- Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide
- Dialogic® IPT Series on Windows® Configuration Guide
- Dialogic® Global Call Country Dependent Parameters (CDP) for PDK Protocols Configuration Guide

3.2.1 Dialogic® System Release 6.1 CompactPCI for Windows® Software Installation Guide

Update to **Chapter 4, Troubleshooting**

In **Section 4.4, Error Messages**, a subsection should be added for the following issue:

Installation of the DetectorsProj Service Fails

A problem has been observed on multiple Windows® 2003 systems where an error occurs during the installation of the Dialogic® System Release Software. A pop-up error message box is displayed when the installation of the DetectorsProj service fails. It has been determined that this error is due to some other software package de-registering the Microsoft® *ATL.DLL* file. This file is not delivered as part of the

Dialogic® System Release Software. If you observe this failure, the following steps can be used to resolve the problem:

1. Change into the Windows System32 directory.

2. Run “regsvr32 atl.dll”.

3. Change into the Dialogic\bin directory.

4. Run “DetectorsProj -service”.

This can be done once the Dialogic® System Release Software installation has completed with the above error, before rebooting. Since this service is not started automatically, the above commands can also be done after reboot, but before the Dialogic® boards are started.

It is only necessary to perform this procedure one time.

3.2.2 Dialogic® DM3 Architecture for CompactPCI on Windows® Configuration Guide

Update to **Section 2.4, Media Loads**

Because of features introduced in the Service Update, new media loads are available for the Dialogic® DM/V2400A-cPCI and DM/V4800BC Media Boards. These media load should be documented in **Section 2.4, Media Loads**. For information about the new media loads, see Section 1.10, “New Media Loads for Dialogic® DM/V4800BC Media Boards Using Special Coders”, on page 36 and Section 1.24, “New Media Loads for Dialogic® DM/V2400A-cPCI and DM/V4800BC Media Boards”, on page 70 of this Release Update.

Update to **Section 2.4.1.1, DM/V, DM/V-A, DM/V-B, and DM/IP Boards**

The following information should be added to indicate that some coders are not supported on the **Dialogic® DM/V4800BC Board** with certain media loads:

When using the Dialogic® DM/V4800BC Board, the maximum bit rate for standard play/record with any basic media load type (for example, media load 1B) is 64 Kbps. The following coders, which are normally part of the standard basic voice media load, are not supported on the DM/V4800BC Board with any basic media load due to density considerations:

- Linear PCM, 8 KHz sampling rate, 16-bit resolution (128 Kbps) VOX and WAVE
- Linear PCM, 11 KHz sampling rate, 8-bit resolution (88 Kbps) VOX and WAVE
- Linear PCM, 11 KHz sampling rate, 16-bit resolution (176 Kbps) VOX and WAVE

Update to **Section 2.4.1.1, DM/V, DM/V-A, DM/V-B, and DM/IP Boards**

In **Table 2, Channel Densities by Board and Media Load (Universal)**, the table footnote about echo cancellation should be changed as follows:

- Default configuration is EEC (enhanced EC, 32 ms) for CSP supported ML, unless otherwise indicated or set in the component named [0x2c] in the respective CONFIG file. You can only change it to a lower EC tail length, by changing the CSP parameter 0x2c03 accordingly in the respective CONFIG file. Conferencing EC, however, will always be 16 ms, regardless of the EC parameter setting.

Update to **Section 3.5, [NFAS] Section**

The third note about NFAS D channel backup (DCBU) supported only on ISDN NI-2 protocol is incorrect. DCBU is supported on 4ESS, 5ESS, and NI-2.

Update to **Section 4.3, Starting the Configuration Manager (DCM)**

In the instructions regarding the Computer Name dialog box for remote DCM, the following note should be added:

Note: In order to use remote DCM, the local computer (management node) and remote computer (managed node) must both be running the same Dialogic® software release (for example, System Release 6.1 CompactPCI Windows) and same build number. If you try to connect to a remote computer with a different software release or build number, the following message is displayed:

```
An incompatible version of the Dialogic(R) System Release
Software is installed on the remote system
```

and the software releases and build numbers for the two systems are shown.

Update to **Section 5.12, Trunk Configuration Property Sheet**

The **Guidelines** at the end of the section (for **Trunk1** to **Trunk16** parameters) should be updated as follows (new information is shown in **bold**):

Guidelines: You can assign different T1 and E1 protocols from the above lists to different trunks on the same board provided the protocols are all from the same group. The following values are for Clear Channel signaling: E1CC, ISDNE1CC, T1CC, and ISDNT1CC. **It is recommended to use ISDNE1CC or ISDNT1CC if mixing ISDN and Clear Channel trunks on the same board. Similarly, use E1CC or T1CC if mixing a PDK (CAS or R2MF) protocol with Clear Channel signaling on the same board.** After assigning T1 or E1 protocols, use the PDK Configuration property sheet to assign country dependent parameter files if applicable.

Update to **Section 6.8, [0x3b] Parameters**

Information about parameters **0x3b03** and **0x3b04** should be added to this section as follows:

Note: This information is intended for experienced users of the DM3 conferencing feature. Changing the default parameter settings is **not** recommended, as this could introduce negative effects on the audio quality and conferencing experience of the participants.

CSUMS_ActTalkerPartiesMinNum

Number: 0x3b03

Description: Specifies the number of talkers in a conference before Active Talker mode is enabled.

Note: Conference Active Talker mode, though related, should not be confused with the Active Talker detection feature.

Values: 0 [default] to 0xff (255).

Guidelines: Refer to the guidelines for the **CSUMS_SmartScalingPartiesMinNum** parameter below.

CSUMS_SmartScalingPartiesMinNum

Number: 0x3b04

Description: Specifies the number of talkers in a conference before scaling mode is enabled.

Values: 0 [default] to 0xff (255).

Guidelines: Audio conferencing provides a mechanism for audio summation of two or more parties in a conference. There are three possible summing modes, which are controlled by CSUMS parameters **0x3b03** and **0x3b04** in the configuration file.

By default, both active talker and scaling are enabled. When parameters **0x3b03** and **0x3b04** are both set to their default values of 0, the default summing mode is Active Talker Summation mode, which sums the three loudest parties. This is advantageous in **large conferences**. Since only the three loudest parties are summed, background noise is reduced. However, there may be times with **small conferences** when a different summation mode is preferable, for example, with soft speakers or when the energy is too low (as with analog lines). The other summation modes are:

- Smart scaling mode - the summation of all parties, but scaling is only done on the ones who are talking.
- No scaling - pure summation, can be used if you want full voice energy in the conference.

The settings for parameters **0x3b03** and **0x3b04** determine the summing mode as shown in the following table.

Parameter 0x3b03, CSUMS_ActTalkerPartiesMinNum	Parameter 0x3b04, CSUMS_SmartScalingPartiesMinNum	Summing Mode
0 (default	0 (default)	Active Talker Detection (default)
> Conf_MaxTotalParties	0 (default)	Smart Scaling
> Conf_MaxTotalParties	> Conf_MaxTotalParties	No Scaling
Conf_MaxTotalParties is the setting for parameter 0x3926 in the configuration file, e.g., SetParm=0x3926,120 !Conf_MaxTotalParties		

To disable the Active Talker algorithm, set the parameter **0x3b03** to a value larger than the maximum number of conferences per DSP; setting it to **Conf_MaxTotalParties**, or per board total number of parties, will suffice, to a maximum of 255.

Even without Active Talker, scaling is also enabled by default. If not required, set the parameter **0x3b04** to a number larger than the maximum number of parties per DSP, and again using **Conf_MaxTotalParties** will suffice, to a maximum of 255.

Update to **Section 6.10, [lineAdmin.x] Parameters (Digital Voice)**

In the guidelines for the **SignalingType** parameter, the note about NFAS D channel backup (DCBU) supported only on ISDN NI-2 protocol is incorrect. DCBU is supported on 4ESS, 5ESS, and NI-2.

Update to **Section 6.21, [CHP] ISDN Protocol Variant Definitions** (IPY00045267)

The values shown for the **ProtocolType** parameter are incorrect. The correct values for the **ProtocolType** parameter are:

- 1: 4ESS
- 2: 5ESS

- 3: DMS100 and DMS250
- 4: NTT
- 7: NET5
- 8: DASS2
- 9: DPNSS
- 10: QSIGE1
- 11: QSIGT1
- 12: NI2

3.2.3 Dialogic® IPT Series on Windows® Configuration Guide

Update to **Section 3.3, Starting the Dialogic® Configuration Manager (DCM)**

In the instructions regarding the Computer Name dialog box for remote DCM, the following note should be added:

Note: In order to use remote DCM, the local computer (management node) and remote computer (managed node) must both be running the same Dialogic® software release (for example, System Release 6.1 CompactPCI Windows) and same build number. If you try to connect to a remote computer with a different software release or build number, the following message is displayed:

```
An incompatible version of the Dialogic(R) System Release
Software is installed on the remote system
```

and the software releases and build numbers for the two systems are shown.

3.2.4 Dialogic® Global Call Country Dependent Parameters (CDP) for PDK Protocols Configuration Guide

Update to **Chapter 10, Brazil R2 Bidirectional Protocol Parameter Configuration**

The **CDP_SKIP_A3_AND_A4_PULSE** parameter should be added to this chapter as follows:

CDP_SKIP_A3_AND_A4_PULSE (Inbound)

Description: Specifies when to send ANI after DNIS when overlap sending is enabled and the Append F flag is disabled. This parameter is valid only if **CDP_OVERLAP_SENDING_ENABLED=1** and **CDP_FLAG_APPEND_F=0**.

Values:

- 0 [default]: Protocol waits for A3 or A4 pulse and then requests Category before requesting ANI. This is the default behavior when **CDP_OVERLAP_SENDING_ENABLED=1** and **CDP_FLAG_APPEND_F=0**.
- 1: Protocol requests ANI immediately after DNIS, without waiting for A3 or A4.

Update to **Chapter 10, Brazil R2 Bidirectional Protocol Parameter Configuration**

The "Values" section for the **CDP_FLAG_APPEND_F** parameter should be updated to refer to the **CDP_SKIP_A3_AND_A4_PULSE** parameter as follows:

Values:

- 0 [default]: No tone will be sent to the remote end. In this case, A3 or A4 pulse is expected to be received from the remote end. In a case of overlapped sending (see description of **CDP_OVERLAP_SENDING_ENABLED** parameter), the remote end may also send A1 to request more information.
  To skip the A3 or A4 pulse and send ANI immediately after DNIS, set **CDP_SKIP_A3_AND_A4_PULSE=1**.
- 1: 'f' (I-15) will be sent to the remote end, indicating the end of information.

3.3 OA&M Documentation Updates

This section contains updates to the following documents (click the title to jump to the corresponding section):

- Dialogic® System Release 6.1 CompactPCI for Windows® Administration Guide
- Dialogic® SNMP Agent Software for Windows® Administration Guide
- Dialogic® System Software Diagnostics Guide
- Dialogic® Board Management API Library Reference
- Dialogic® Event Service API Programming Guide
- Dialogic® Event Service API Library Reference
- Dialogic® Native Configuration Manager API Programming Guide
- Dialogic® Native Configuration Manager API Library Reference

3.3.1 Dialogic® System Release 6.1 CompactPCI for Windows® Administration Guide

Update to **Chapter 4, Administrative Utilities**

The following procedures for checking the firmware version and upgrading the firmware for Dialogic® IPT Boards should be added.

IPT Board Firmware Update Utility

The IPT Board Firmware Update utility facilitates the updating of the firmware residing in flash memory on a Dialogic® IPT Board. In order to perform this procedure, the board needs to be powered up and properly displayed by the Dialogic® Configuration Manager (DCM).

First, to determine the version of firmware running on the IPT Board:

1. In the DCM, select the IPT Board to upgrade, right-click on it, and choose **Configure Device**.
2. Go to the **Software Constants** property sheet.

3. Check the value of the **IPTBoardSoftwareVersion** parameter, which has a hex value that represents the firmware version. For example, the parameter value "10a0918" translates to firmware version NDB_1.10.9_B24; the last 2 digits represent the build number (i.e., hex 18 is 24 decimal).

To determine if a newer firmware version is available in the Dialogic® System Release 6.1 CompactPCI for Windows® Service Update baseline:

1. Open up a DOS CMD prompt.
2. Go to the *..\dialogic\data* directory.
3. Run "`getver pmac_stl_672.bin`".
   This returns the following output, for example:
   ```
   Processing file pmac_stl_672.bin
   Embedded name: PMAC_binary
   Version:       1.10  Build: NBD_1.10.9_B24
   Done
   ```

If you determine that an upgrade is necessary, upgrade the IPT Board firmware as follows:

1. Make sure the Dialogic® services are stopped through the DCM.
2. Open up a DOS CMD prompt.
3. Go to the local directory that contains the *pmac_stl_672.bin* file to be used for the upgrade, in this case, *..\dialogic\data*.
4. Run "`pmacadmin -l`" to determine the board numbers of the IPT Boards in your system.
5. Execute the following command to hard reset the board:
   "`pmacadmin -e -P #`"
   (where # is the physical slot in which the board resides in the chassis).
6. Execute the following command to upgrade the firmware on the IPT Board:
   "`dlgpmacfwupdate.exe -f pmac_stl_672.bin -b # -t 1`"
   (where # is the board number to be upgraded).
7. Restart the driver and firmware running on the board; this can be done by just rebooting the system.

3.3.2 Dialogic® SNMP Agent Software for Windows® Administration Guide

There are currently no updates to this document.

3.3.3 Dialogic® System Software Diagnostics Guide

Update for Remote Diagnostics Package

A remote diagnostics package is now available that allows you to run Dialogic® diagnostics utilities remotely from a central site. For further information, see Section 1.17, "Remote Diagnostics Package", on page 58 of this Release Update.

Update to **Chapter 12, DebugAngel Reference**

Because of a new feature in the Service Update, the DebugAngel tool has been enhanced to provide more capabilities for managing multiple log files. For more information about this feature, see Section 1.11, “File Management Enhancements for DebugAngel Tool”, on page 40 of this Release Update.

Update to **Chapter 20, ISDN Trace Reference**

Because of a new feature in the Service Update, the ISDNtrace tool has been enhanced to include new command line options to set the output log file size and to create multiple backup log files to be archived. For more information about this feature, see Section 1.23, “File Management Enhancements for ISDNtrace Tool”, on page 66 of this Release Update.

Update to **Chapter 21, Telecom Subsystem Summary Tool Reference**

Because of an enhancement in the Service Update, the its_sysinfo.htm file now includes a Windows Package Info section at the beginning of the file. For further information about this feature, see Section 1.26, “Enhancement to its_sysinfo Tool”, on page 75 of this Release Update.

Update to **Chapter 24, PDK Trace Reference**

Because of a new feature in the Service Update, PDK Trace has new command line options to set the output log file size and to create multiple log files. For more information about this feature, see Section 1.12, “File Management Enhancements for PDK Trace Tool”, on page 45 of this Release Update.

Update to **Chapter 26, PSTN Diagnostics Tool Reference**

An enhanced version of the PSTN Diagnostics tool (pstndiag) is provided in the Service Update. The previous version of the tool is still supported. For information about the new version, see Section 1.18, “Enhanced Diagnostics Tools”, on page 59 of this Release Update.

Update to **Chapter 28, Runtime Trace Facility (RTF) Reference** (IPY00037518)

The following information about using binary log files should be added to **Section 28.3.2, Logfile Tag**:

For installations with high channel densities, or which have enabled all or most RTF trace levels, the volume of logging may result in an increased CPU utilization by the RtfServer executable as a result of the increased volume of log messages.

As shipped, the RTF log files are generated in ASCII text mode. There is a configuration parameter in the RTF configuration file (*RtfConfigWin.xml* for Windows, *RtfConfigLinux.xml* for Linux) that allows log files to be generated in either “text” or “binary” format. Testing on high channel density systems with most or all of the RTF trace levels enabled has shown that the generation of binary format RTF log files has less of an impact on CPU usage than does the generation of text format RTF log files.

If the volume of logging results in high CPU usage, then using binary format will reduce the usage.

Enabling Binary Format RTF Log Files

The XML file contains the following line, which allows changes to log file parameters to be made:

```
<Logfile path="$(INTEL_DIALOGIC_DIR)\log" size="300" maxbackups="10"
preserve_size="300" preserve_maxbackups="10"
duplicate_to_debug_console="0" log_format="text" />
```

The “log_format” value controls the type of log files that are written. Valid values for this parameter are “text” and “binary”. Once a change has been made to the XML file, it must be reloaded using the rtftool reload command.

Converting Binary Format RTF Log Files to Text Format

In order for binary log files to be examined, they must be converted into text format. This can be done by using the rtftool export command.

```
rtftool export [-d source_dir | -s source_file]
[-f [dest_file] | -m dest_dir]
```

By default, the name of the text format files generated by this command will be *EXPORT-<RTF binary log file name>*. For example, if the binary format file is named *rtflog-LOCAL-20070306-15h09m26.506s.txt*, then the default name of the generated text format file will be *EXPORT-rtflog-LOCAL-20070306-15h09m26.506s.txt.* This behavior can be overridden using the -f command line option.

The rtftool utility is a stand-alone program, and it is not necessary to have the Dialogic System Release installed on the system in order to convert RTF log files from binary to text format.

Note: When generating large binary files with RTF, do not split the single large binary file and then use the individual split files with the rtftool utility. Rtftool will not work with chopped binary files.

Update to **Chapter 29, RTFManager Reference** (IPY00037518)

Section 29.5, General Tab, says that binary log format is not supported by the current release. This is not correct; binary log format is supported. For information about binary log files, see the update to **Chapter 28, Runtime Trace Facility (RTF) Reference** above.

Update to **Chapter 30, Status Monitor Reference**

An enhanced version of the Status Monitor tool (statusmon) is provided in the Service Update. The previous version of the tool is still supported. For information about the new version, see Section 1.18, “Enhanced Diagnostics Tools”, on page 59 of this Release Update.

3.3.4 Dialogic® Board Management API Library Reference

There are currently no updates to this document.

3.3.5 Dialogic® Event Service API Programming Guide

There are currently no updates to this document.

3.3.6 Dialogic® Event Service API Library Reference

There are currently no updates to this document.

3.3.7 Dialogic® Native Configuration Manager API Programming Guide

There are currently no updates to this document.

3.3.8 Dialogic® Native Configuration Manager API Library Reference

There are currently no updates to this document.

3.4 Programming Libraries Documentation Updates

This section contains updates to the following documents (click the title to jump to the corresponding section):

- Dialogic® Audio Conferencing API Programming Guide
- Dialogic® Audio Conferencing API Library Reference
- Dialogic® Continuous Speech Processing API Programming Guide
- Dialogic® Continuous Speech Processing API Library Reference
- Dialogic® Digital Network Interface Software Reference
- Dialogic® Fax Software Reference
- Dialogic® Global Call API Programming Guide
- Dialogic® Global Call API Library Reference
- Dialogic® Global Call E1/T1 CAS/R2 Technology Guide
- Dialogic® Global Call IP Technology Guide
- Dialogic® Global Call ISDN Technology Guide
- Dialogic® Global Call SS7 Technology Guide
- Dialogic® IP Media Library API Programming Guide
- Dialogic® IP Media Library API Library Reference
- Dialogic® Modular Station Interface API Programming Guide
- Dialogic® Modular Station Interface API Library Reference
- Dialogic® Standard Runtime Library API Programming Guide
- Dialogic® Standard Runtime Library API Library Reference
- Dialogic® Voice API Programming Guide
- Dialogic® Voice API Library Reference

3.4.1 Dialogic® Audio Conferencing API Programming Guide

There are currently no updates to this document.

3.4.2 Dialogic® Audio Conferencing API Library Reference

There are currently no updates to this document.

3.4.3 Dialogic® Continuous Speech Processing API Programming Guide

There are currently no updates to this document.

3.4.4 Dialogic® Continuous Speech Processing API Library Reference

Update to **ec_reciottdata( )** and **ec_stream( )**

The **ec_reciottdata( )** and **ec_stream( )** function reference pages contain a caution about channels getting stuck when failing to listen to a TDM bus time slot prior to invoking a record operation. The caution should be revised: this condition now returns an error rather than resulting in a stuck channel. The revised caution is:

- On Dialogic® DM3 Boards using a flexible routing configuration, CSP channels must be listening to a TDM bus time slot in order for the **ec_reciottdata( )** and **ec_stream( )** functions to work. The actual recording operation will start only after the channel is listening to the proper external time slot. In other words, you must issue a **dx_listen( )** function call on the device handle before calling **ec_reciottdata( )** or **ec_stream( )** for that device handle, and the **dx_listen( )** has to be called from the same process as the **ec_reciottdata( )** or **ec_stream( )**. If not, the **ec_reciottdata( )** or **ec_stream( )** function will return TEC_STREAM with EDX_SH_MISSING as the termination reason.

3.4.5 Dialogic® Digital Network Interface Software Reference

There are currently no updates to this document.

3.4.6 Dialogic® Fax Software Reference

There are currently no updates to this document.

3.4.7 Dialogic® Global Call API Programming Guide

There are currently no updates to this document.

3.4.8 Dialogic® Global Call API Library Reference

Update to **gc_GetFrame( )**

In the code example, under `/* Retrieve events from SRL */`

change this:

```
GC_L2_BLK recvfrmptr; /* Buffer to store received frame */
.
.
.
case GCEV_L2FRAME:
/* retrieve signaling information from queue */
if ( gc_GetFrame(ldev, &recvfrmptr) != GC_SUCCESS)
```

to this:

```
L2_BLK        l2ie;
GC_L2_BLK     recvfrmptr; /* Buffer to store received frame */
.
.
.
recvfrmptr.cclib = (void *)&l2ie;
case GCEV_L2FRAME:
/* retrieve signaling information from queue */
if ( gc_GetFrame(ldev, &recvfrmptr) != GC_SUCCESS)
```

Update to **gc_util_insert_parm_val( )** (IPY00043078)

In the description for **gc_util_insert_parm_val( )**, a note should be added stating that **gc_Start( )** must be called before **gc_util_insert_parm_val( )**. Also, the code example should be replaced with the following:

```
#include <stdio.h>
#include <srllib.h>
#include <gclib.h>
#include <gcerr.h>

void main( )
{
    GC_PARM_BLKP my_blkp = NULL;
    GC_PARM_DATAP my_parmp;
    GC_INFO gc_error_info; /* GlobalCall error information data */
    int type = 1;

    /* Issue a gc_Start() call to initialize the library */
    if ( gc_Start(NULL) != GC_SUCCESS )
    {
        /* process error return as shown */
        gc_ErrorInfo( &gc_error_info );
        printf ("Error: gc_Start(), GC ErrorValue: 0x%hx - %s, CCLibID: %i - %s,
                CC ErrorValue: 0x%lx - %s\n", gc_error_info.gcValue, gc_error_info.gcMsg,
                gc_error_info.ccLibId, gc_error_info.ccLibName,
                gc_error_info.ccValue, gc_error_info.ccMsg);
        return (gc_error_info.gcValue);
    }

    /* insert parm by reference */
    if ( gc_util_insert_parm_ref( &my_blkp, GC_SET_SERVREQ, PARM_REQTYPE,
                                  sizeof( int ), &type ) != GC_SUCCESS )
    {
        /* Process error */
    }
    /* insert parm by value */
    if ( gc_util_insert_parm_val( &my_blkp, GC_SET_SERVREQ, PARM_ACK,
                                  sizeof( short ), GC_ACK ) != GC_SUCCESS )
    {
        /* Process error */
    }

    /* Now we should have a GC_PARM_BLK with 2 parameters */
```

```
    /* Following use of gc_util_next_parm retrieves the first parameter in a
    * GC_PARM_BLK, which in this case is PARM_REQTYPE */
    my_parmp = gc_util_next_parm( my_blkp, NULL );

    /* Retrieve the next parameter after getting the first one */
    my_parmp = gc_util_next_parm( my_blkp, my_parmp );

    /* This function finds and returns specified parameter, NULL if not found */
    my_parmp = gc_util_find_parm( my_blkp, GC_SET_SERVREQ, PARM_ACK );

    /* After GC_PARM_BLK is no longer needed, delete the block */
    gc_util_delete_parm_blk( my_blkp );

    /* Set my_blkp to NULL now that the block has been deleted */
    my_blkp = NULL;

    /* Issue gc_Stop() Next */
    if (gc_Stop() != GC_SUCCESS )
    {
        /* process error return as shown */
        gc_ErrorInfo( &gc_error_info );
        printf ("Error: gc_Stop(), GC ErrorValue: 0x%hx - %s, CCLibID: %i - %s,
                CC ErrorValue: 0x%lx - %s\n",
                gc_error_info.gcValue, gc_error_info.gcMsg,
                gc_error_info.ccLibId, gc_error_info.ccLibName,
                gc_error_info.ccValue, gc_error_info.ccMsg);
    }
}
```

3.4.9 Dialogic® Global Call E1/T1 CAS/R2 Technology Guide

There are currently no updates to this document.

3.4.10 Dialogic® Global Call IP Technology Guide

Updates for new QoS alarms

With the Service Update, two new Quality of Service (QoS) alarms are supported on Dialogic® IPT Boards:

- QOSTYPE_RTCPTIMEOUT - QoS alarm for Real Time Control Protocol (RTCP) inactivity
- QOSTYPE_RTPTIMEOUT - QoS alarm for Real Time Protocol (RTP) inactivity

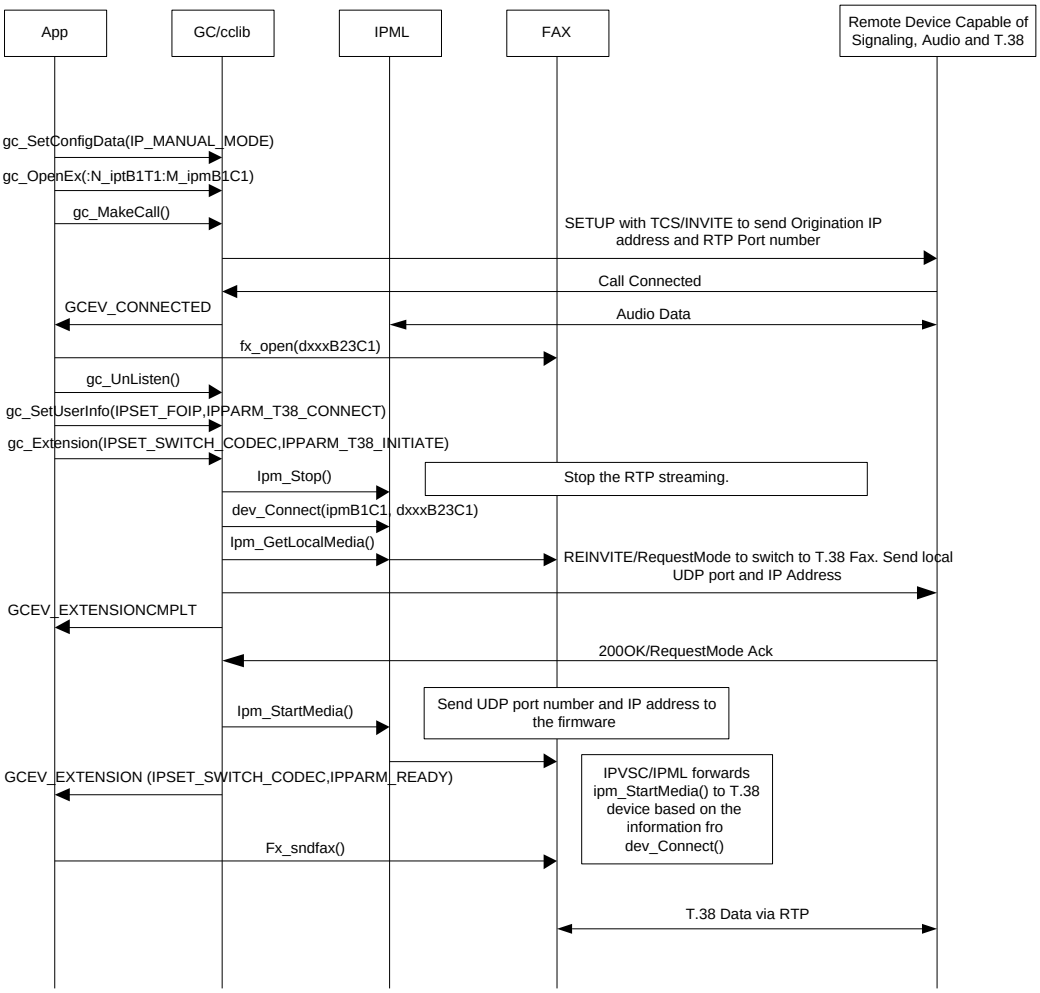
These alarms and their associated events should be included in the following sections of the *Dialogic® Global Call IP Technology Guide*:

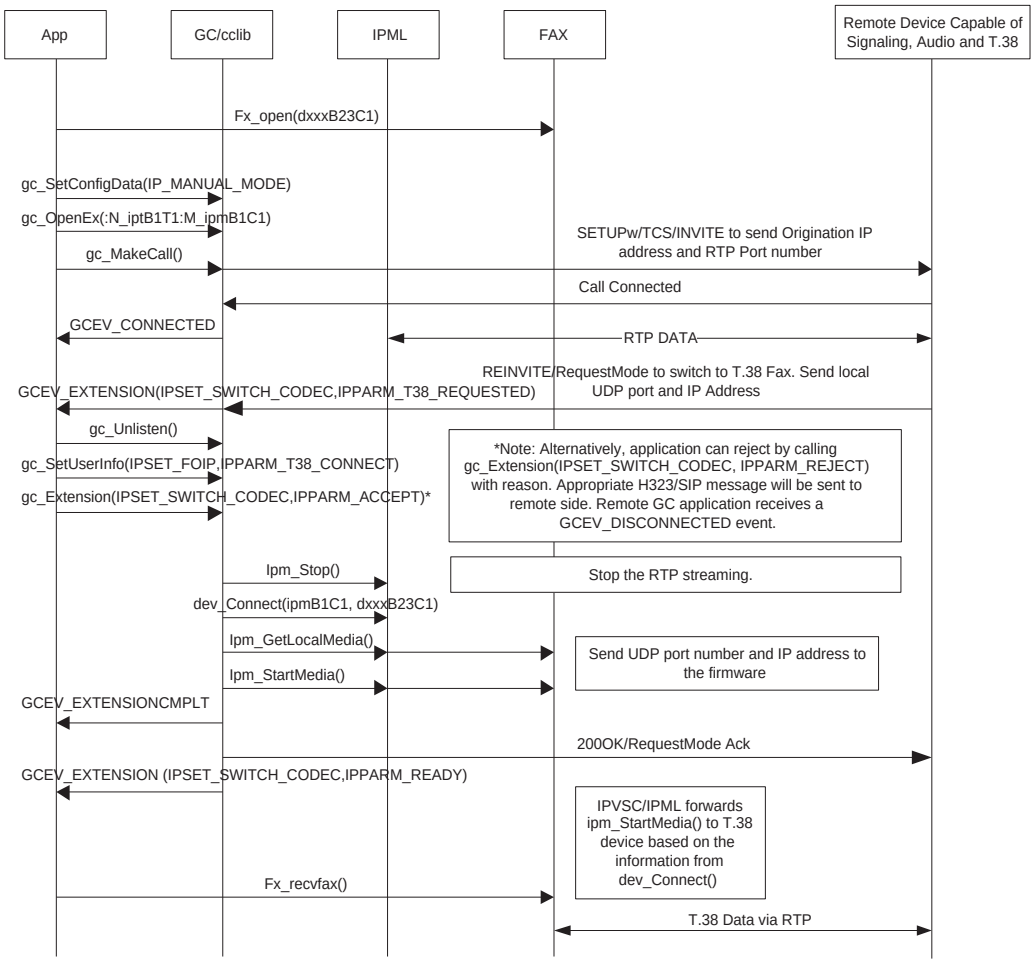
- **Section 4.20, Managing Quality of Service Alarms**
- **Section 7.3.9, gc_GetAlarmParm( ) Variances for IP**
- **Section 7.3.24, gc_SetAlarmParm( ) Variances for IP**

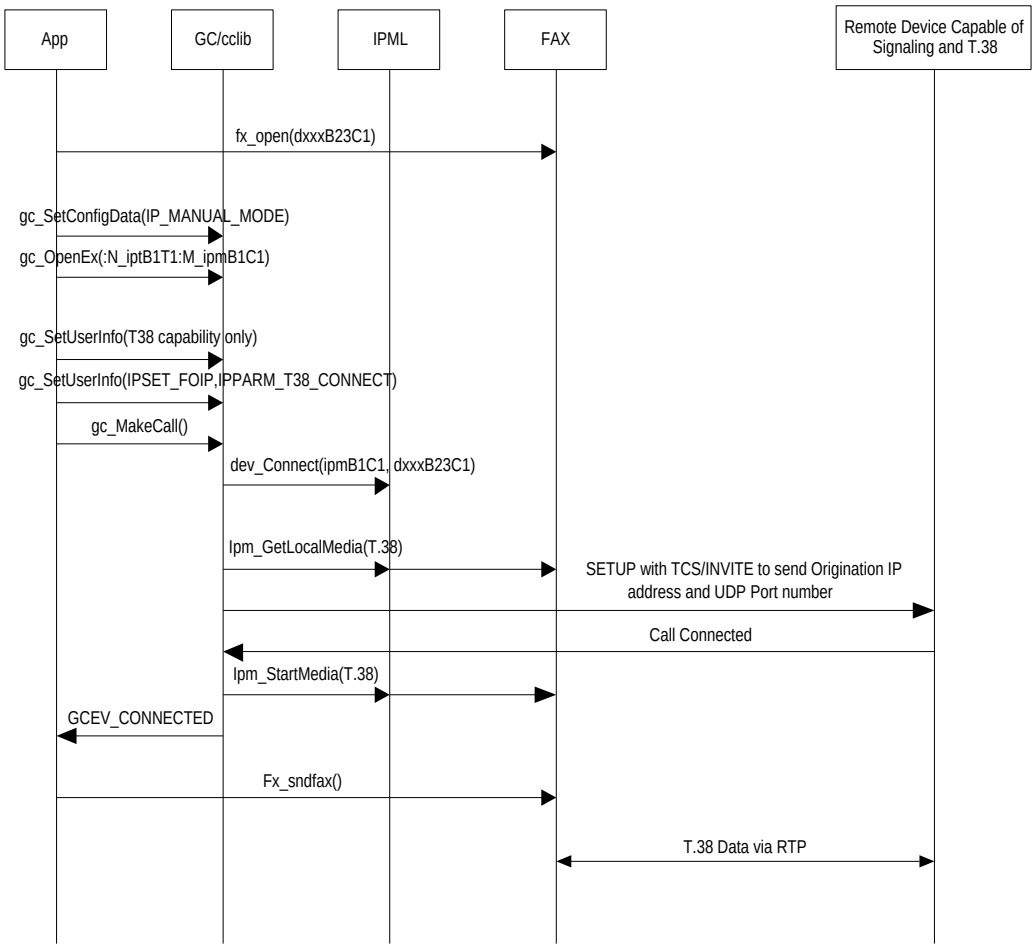
For further information about these alarms, see Section 1.19, “New QoS Alarms for RTCP and RTP Inactivity”, on page 61 of this Release Update.

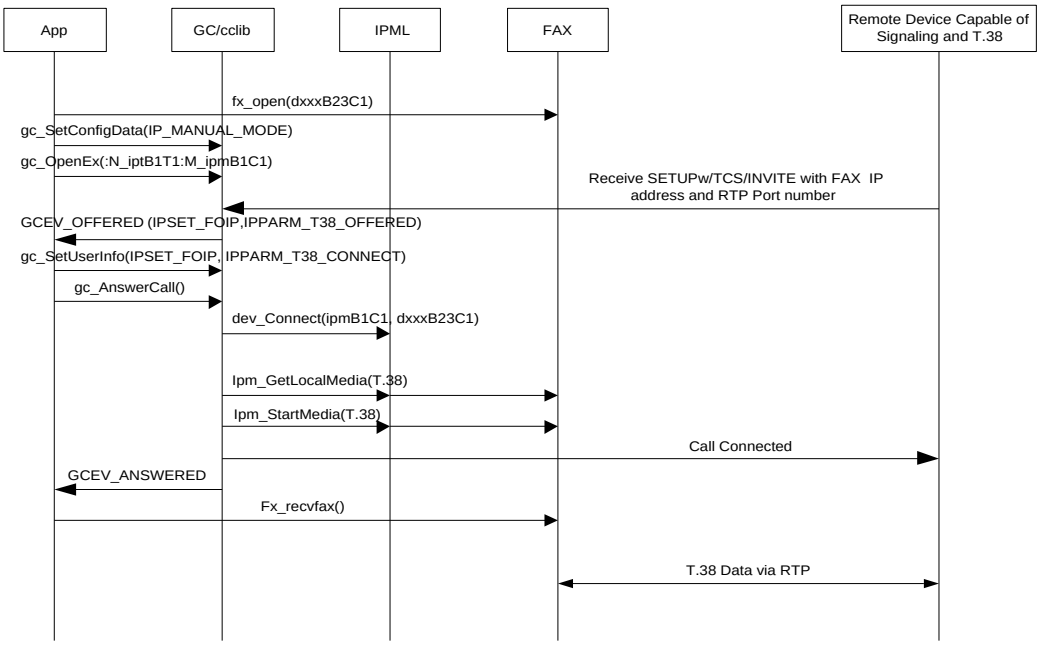
Update to **Chapter 3. IP Call Scenarios**

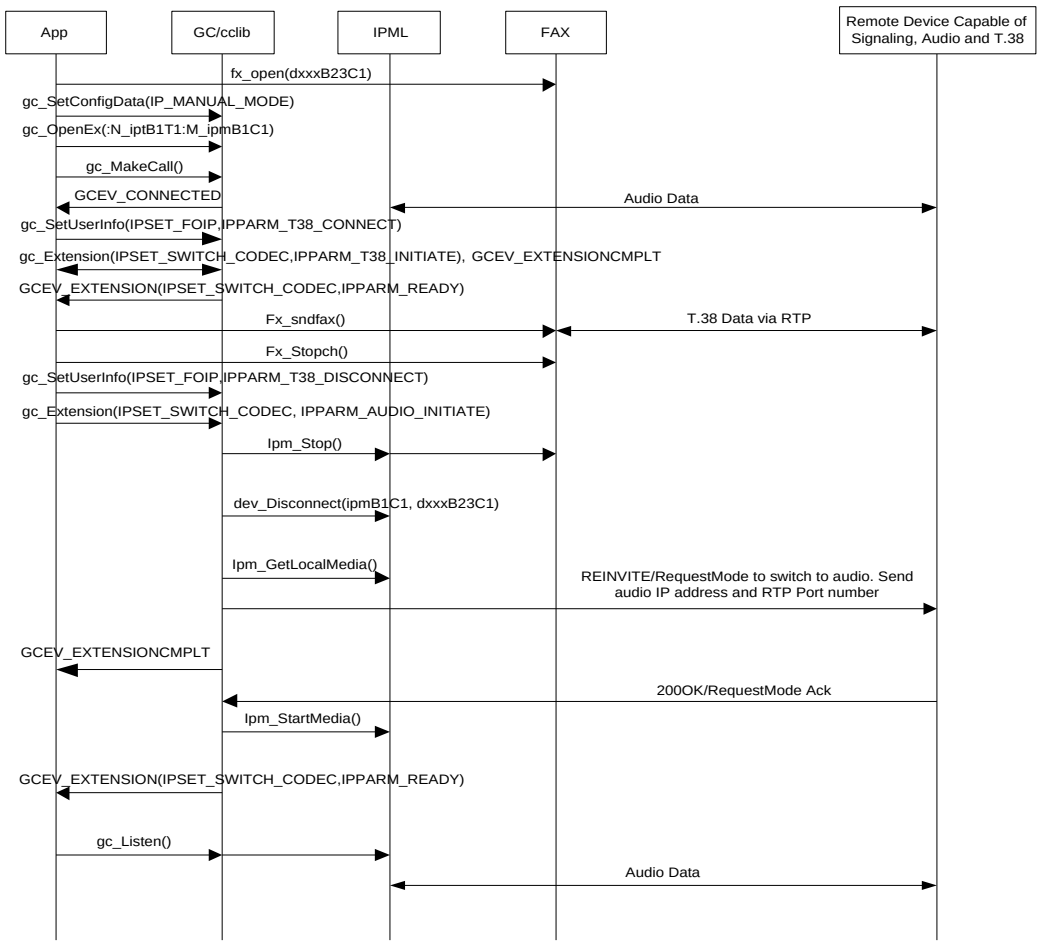
Add new section, T.38 The following figure

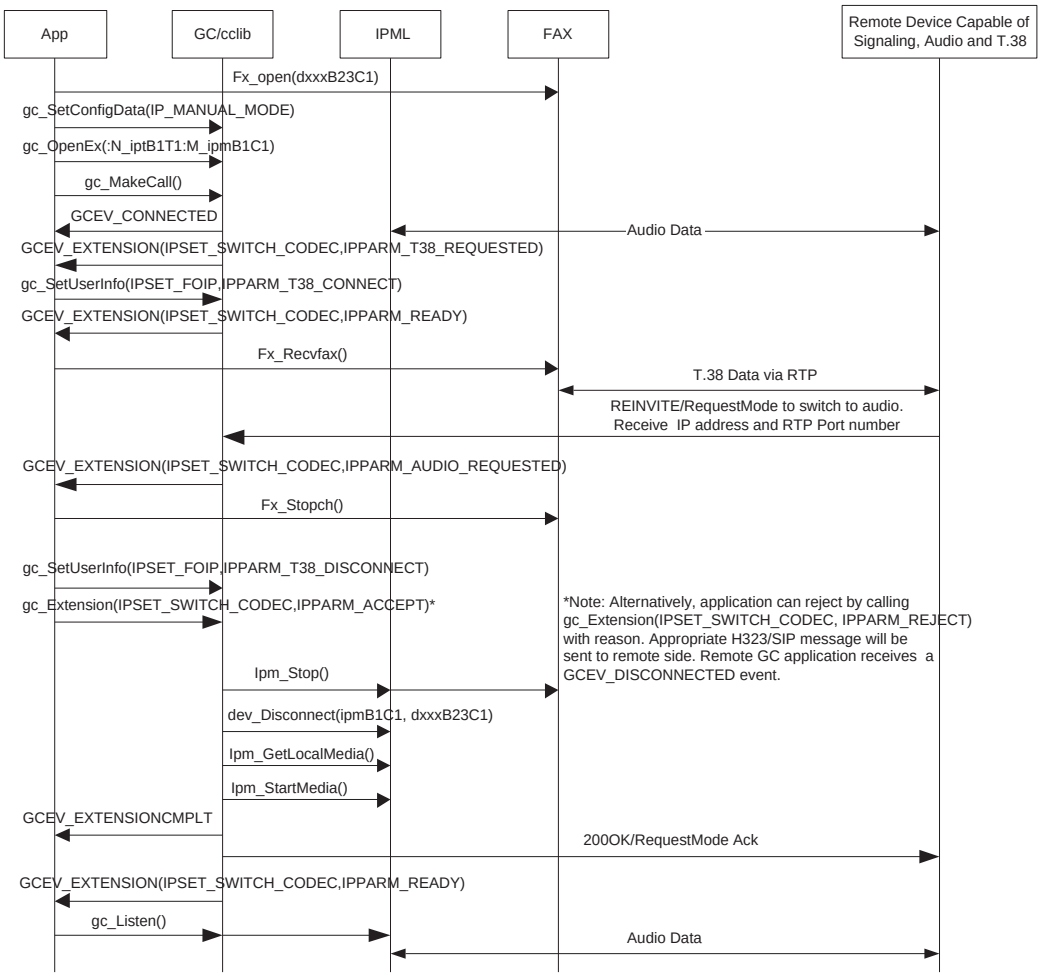


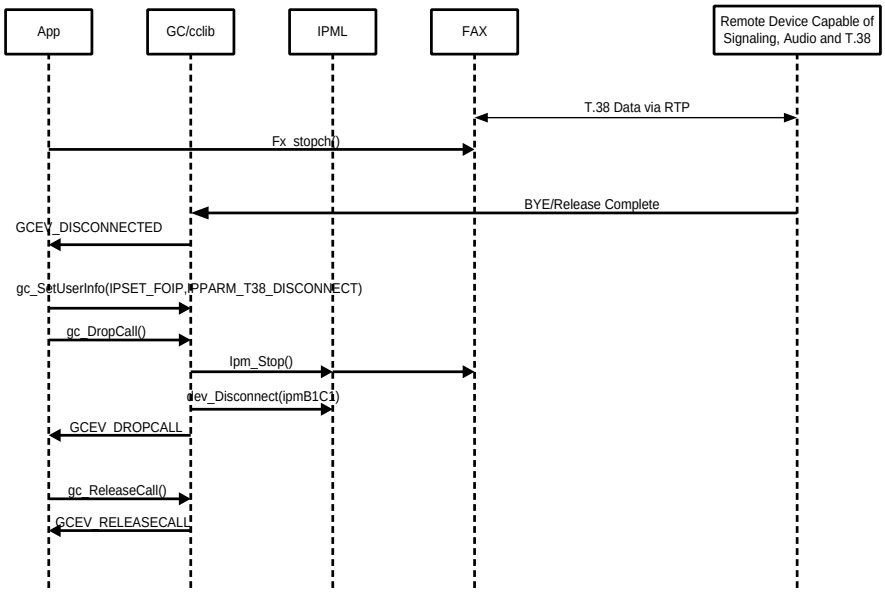

App
GC/cclib
IPML
FAX
Remote Device Capable of Signaling, Audio and T.38
Fx_open(dxxxB23C1)
gc_SetConfigData(IP_MANUAL_MODE)
gc_OpenEx(:N_iptB1T1:M_ipmB1C1)
gc_MakeCall()
SETUPw/TCS/INVITE to send Origination IP address and RTP Port number
Call Connected
GCEV_CONNECTED
RTP DATA
REINVITE/RequestMode to switch to T.38 Fax. Send local UDP port and IP Address
GCEV_EXTENSION(IPSET_SWITCH_CODEC,IPPARM_T38_REQUESTED)
gc_Unlisten()
gc_SetUserInfo(IPSET_FOIP, IPPARM_T38_CONNECT)
gc_Extension(IPSET_SWITCH_CODEC,IPPARM_ACCEPT)*
*Note: Alternatively, application can reject by calling gc_Extension(IPSET_SWITCH_CODEC, IPPARM_REJECT) with reason. Appropriate H323/SIP message will be sent to remote side. Remote GC application receives a GCEV_DISCONNECTED event.
Ipm_Stop()
Stop the RTP streaming.
dev_Connect(ipmB1C1, dxxxB23C1)
Ipm_GetLocalMedia()
Send UDP port number and IP address to the firmware
Ipm_StartMedia()
GCEV_EXTENSIONCMPLT
200OK/RequestMode Ack
GCEV_EXTENSION (IPSET_SWITCH_CODEC,IPPARM_READY)
IPVSC/IPML forwards ipm_StartMedia() to T.38 device based on the information from dev_Connect()
Fx_recvfax()
T.38 Data via RTP


App
GC/cclib
IPML
FAX
Remote Device Capable of Signaling and T.38
fx_open(dxxxB23C1)
gc_SetConfigData(IP_MANUAL_MODE)
gc_OpenEx(:N_iptB1T1:M_ipmB1C1)
gc_SetUserInfo(T38 capability only)
gc_SetUserInfo(IPSET_FOIP,IPPARM_T38_CONNECT)
gc_MakeCall()
dev_Connect(ipmB1C1, dxxxB23C1)
Ipm_GetLocalMedia(T.38)
SETUP with TCS/INVITE to send Origination IP address and UDP Port number
Call Connected
Ipm_StartMedia(T.38)
GCEV_CONNECTED
Fx_sndfax()
T.38 Data via RTP


App
GC/cclib
IPML
FAX
Remote Device Capable of Signaling and T.38
fx_open(dxxxB23C1)
gc_SetConfigData(IP_MANUAL_MODE)
gc_OpenEx(:N_iptB1T1:M_ipmB1C1)
Receive SETUPw/TCS/INVITE with FAX IP address and RTP Port number
GCEV_OFFERED (IPSET_FOIP,IPPARM_T38_OFFERED)
gc_SetUserInfo(IPSET_FOIP, IPPARM_T38_CONNECT)
gc_AnswerCall()
dev_Connect(ipmB1C1, dxxxB23C1)
Ipm_GetLocalMedia(T.38)
Ipm_StartMedia(T.38)
Call Connected
GCEV_ANSWERED
Fx_recvfax()
T.38 Data via RTP

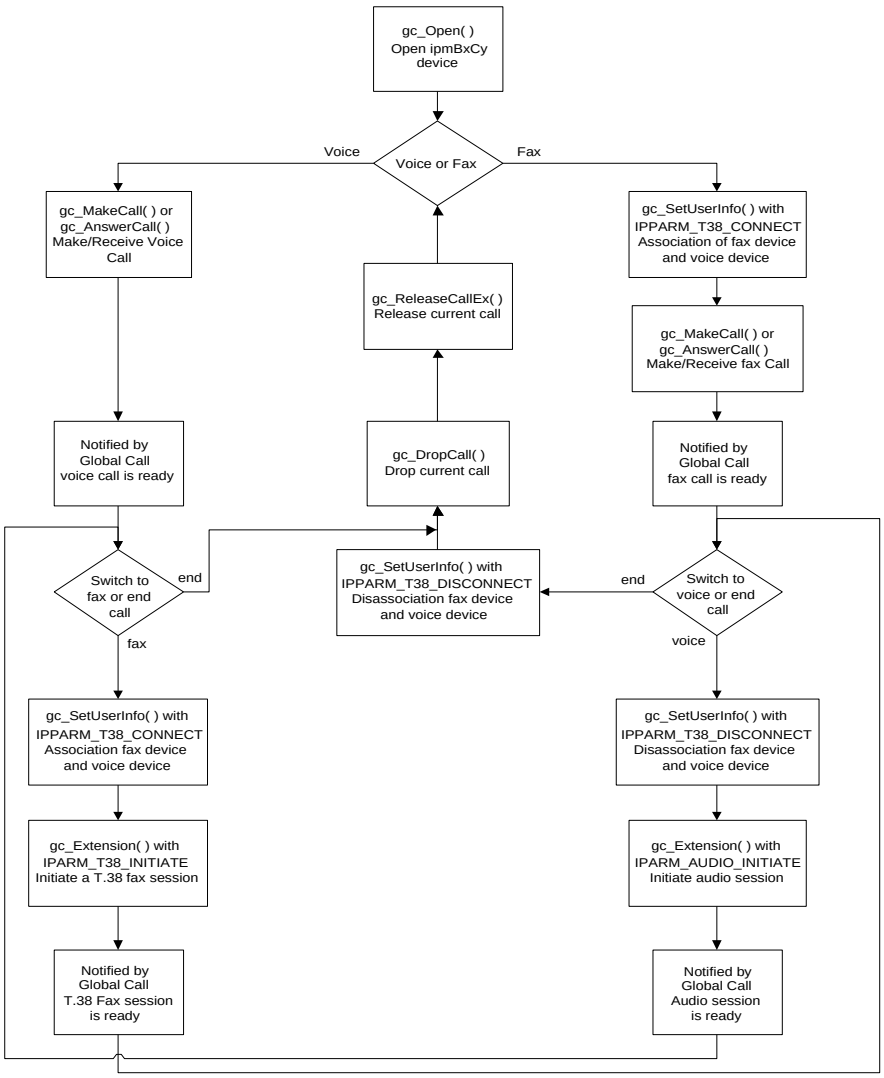

App
GC/cclib
IPML
FAX
Remote Device Capable of Signaling, Audio and T.38
fx_open(dxxxB23C1)
gc_SetConfigData(IP_MANUAL_MODE)
gc_OpenEx(:N_iptB1T1:M_ipmB1C1)
gc_MakeCall()
GCEV_CONNECTED
Audio Data
gc_SetUserInfo(IPSET_FOIP,IPPARM_T38_CONNECT)
gc_Extension(IPSET_SWITCH_CODEC,IPPARM_T38_INITIATE), GCEV_EXTENSIONCMPLT
GCEV_EXTENSION(IPSET_SWITCH_CODEC,IPPARM_READY)
Fx_sndfax()
T.38 Data via RTP
Fx_Stopch()
gc_SetUserInfo(IPSET_FOIP,IPPARM_T38_DISCONNECT)
gc_Extension(IPSET_SWITCH_CODEC, IPPARM_AUDIO_INITIATE)
Ipm_Stop()
dev_Disconnect(ipmB1C1, dxxxB23C1)
Ipm_GetLocalMedia()
REINVITE/RequestMode to switch to audio. Send audio IP address and RTP Port number
GCEV_EXTENSIONCMPLT
200OK/RequestMode Ack
Ipm_StartMedia()
GCEV_EXTENSION(IPSET_SWITCH_CODEC,IPPARM_READY)
gc_Listen()
Audio Data


App
GC/cclib
IPML
FAX
Remote Device Capable of Signaling, Audio and T.38
Fx_open(dxxxB23C1)
gc_SetConfigData(IP_MANUAL_MODE)
gc_OpenEx(:N_iptB1T1:M_ipmB1C1)
gc_MakeCall()
GCEV_CONNECTED
Audio Data
GCEV_EXTENSION(IPSET_SWITCH_CODEC,IPPARM_T38_REQUESTED)
gc_SetUserInfo(IPSET_FOIP,IPPARM_T38_CONNECT)
GCEV_EXTENSION(IPSET_SWITCH_CODEC,IPPARM_READY)
Fx_Recvfax()
T.38 Data via RTP
REINVITE/RequestMode to switch to audio.
Receive IP address and RTP Port number
GCEV_EXTENSION(IPSET_SWITCH_CODEC,IPPARM_AUDIO_REQUESTED)
Fx_Stopch()
gc_SetUserInfo(IPSET_FOIP,IPPARM_T38_DISCONNECT)
gc_Extension(IPSET_SWITCH_CODEC,IPPARM_ACCEPT)*
*Note: Alternatively, application can reject by calling gc_Extension(IPSET_SWITCH_CODEC, IPPARM_REJECT) with reason. Appropriate H323/SIP message will be sent to remote side. Remote GC application receives a GCEV_DISCONNECTED event.
Ipm_Stop()
dev_Disconnect(ipmB1C1, dxxxB23C1)
Ipm_GetLocalMedia()
Ipm_StartMedia()
GCEV_EXTENSIONCMPLT
200OK/RequestMode Ack
GCEV_EXTENSION(IPSET_SWITCH_CODEC,IPPARM_READY)
gc_Listen()
Audio Data



Update to Chapter 4. IP Specific Operations

Add a new section, T.38 Fax Server, along with the following information:



Update to **Section 4.7.5, Responding to SIP re-INVITE Requests**

With the Service Update, the information in this section is now applicable to Dialogic® IPT Boards. References designating that the feature is supported on “DM/IP only” should be deleted. For further information, see Section 1.27, “Modify an Existing SIP Call Using re-INVITE for Dialogic® IPT Boards”, on page 76 in this Release Update.

Update to **Section 4.14, Sending and Receiving SIP OPTIONS Messages**

The INFO method is included as part of the Allow header in SIP messages by default. Section 4.14 should be updated in three places:

- Under **Section 4.14.1, Default OPTIONS Behavior**, change as follows:
  The default Allow header will be the following if supplementary services (call transfer) is not enabled:
  Allow: INVITE, CANCEL, ACK, BYE, **INFO**
  or the following if supplementary services is enabled:

Allow: INVITE, CANCEL, ACK, BYE, **INFO**, REFER, NOTIFY
Note that in either case, OPTIONS is not included in the list.

- Under **Section 4.14.3, Sending OPTIONS Requests**, change as follows:
  When SIP OPTIONS access is enabled, the Allow header field will be the following if supplementary services (call transfer) is not enabled:
  Allow: INVITE, CANCEL, ACK, BYE, **INFO**, OPTIONS
  or the following if supplementary services is enabled:
  Allow: INVITE, CANCEL, ACK, BYE, **INFO**, REFER, NOTIFY, OPTIONS
  The application can add additional methods to the Allow header, but the Global Call library will ensure that all of the methods supported by the library are included.
- Under **Section 4.14.6, Responding to OPTIONS Requests**, in the **"Success" Response Message** subsection, change as follows:
  The Global Call library ensures that the Allow header field contains all SIP methods supported by the library, which includes the following methods if supplementary services (call transfer) is not enabled:
  INVITE, CANCEL, ACK, BYE, **INFO**, OPTIONS
  or the following if supplementary services is enabled:
  INVITE, CANCEL, ACK, BYE, **INFO**, REFER, NOTIFY, OPTIONS

Update to **Section 4.24.2, Initiating Fax Transcoding** (IPY00040070)

The following note should be added in this section:

Note: CNG tone detection must be specifically enabled by using **gc_SetConfigData( )**, where the set ID is IPSET_TDM_TONEDET and the parameter ID is IPPARM_TDMDET_CNG_ENABLE.

See the related documentation updates for **Section 8.1, Overview of Parameter Usage**, and **Section 8.2.26, IPSET_TDM_TONEDET**, for additional information about IPPARM_TDMDET_CNG_ENABLE and IPPARM_TDMDET_CNG_DISABLE.

Update to **Section 4.26.2, Media LAN Disconnection Alarm**

Because of a new feature in the Service Update, this section should be updated to indicate that it is applicable to Dialogic® DM/IP601-CPCI-100BT and Dialogic® DM/IP601-2E1-CPCI-100BT IP Boards, as well as to Dialogic® IPT Boards. For further information, see Section 1.8, "Media LAN Disconnection Alarm Notification for Dialogic® DM/IP Boards", on page 33 of this Release Update.

Update to **Section 8.1, Overview of Parameter Usage** (IPY00040070)

In **Table 34, Summary of Parameter Sets and Parameter Usage**, parameter IDs IPPARM_TDMDET_CNG_ENABLE and IPPARM_TDMDET_CNG_DISABLE should be added for IPSET_TDM_TONEDET as follows:

Set ID	Parameter ID	Set	Send	Retrieve	SIP/ H.323
IPSET_ TDM_TONEDET	IPPARM_ TDMDET_CNG_ ENABLE	**gc_SetConfigData( )**	---	---	both
	IPPARM_ TDMDET_CNG_ DISABLE	**gc_SetConfigData( )**	---	---	both

Update to **Section 8.2, Parameter Set Reference**

New parameter IDs have been added to IPSET_CALLINFO so that you can send and receive CPN fields via Global Call over an IP network. See Section 1.30, "Dialogic® Global Call API Access to New H.323/Q.931 Message IEs", on page 83 of this Release Update for more details.

Update to **Section 8.2.4, IPSET_CONFIG**

New parameter IDs have been added to IPSET_CONFIG so that you can disable/enable the sending of the automatic SIP re-INVITE message upon media switch (e.g., when switching from fax to audio). For further information, see Section 1.14, "Disabling Automatic re-INVITE Message when Switching between Fax and Audio", on page 51 of this Release Update.

Update to **Section 8.2.26, IPSET_TDM_TONEDET** (IPY00040070)

In **Table 60, IPSET_TDM_TONEDET Parameter Set**, parameter IDs IPPARM_TDMDET_CNG_ENABLE and IPPARM_TDMDET_CNG_DISABLE should be added as follows:

Parameter IDs	Type & Size	Description	SIP/ H.323
IPPARM_TDMDET_CNG_ENABLE	Type: char Size: 1 byte Value: 1	Enables CNG tone detection on TDM for IP session switching to fax transcoding	both
IPPARM_TDMDET_CNG_DISABLE	Type: char Size: 1 byte Value: 1	Disables CNG tone detection on TDM for IP session switching to fax transcoding	both

Update to **Chapter 9, IP-Specific Data Structures**

Because of a feature introduced in the Service Update, a new data structure, SIP_STACK_CFG, has been added for configuring SIP stack parameters. Related to this, the IP_VIRTBOARD data structure has been updated to point to the new structure. For further information, see Section 1.13, "Configuring SIP Stack Parameters with Global Call", on page 47 of this Release Update.

Updates to **Chapter 11, IP-Specific Event Cause Codes** (IPY00081301)

With Service Update 193, a new SIP error code is added to *gcip_defs.h* header file. This error is can be retrieved upon receipt of GCEV_TASKFAIL and has the following GC_INFO values:

gcValue: EGC_CCLIBSPECIFIC
ccLibId: GC_H3R_LIB
ccValue: IPEC_InternalHardwareFailure

This new error code is:

IPEC_InternalHardwareFailure = 5804 /* 0x16ac */
An internal IPT device hardware malfunction prevented completion of the IP call control library message on the channel.

This error code is added to the following sections:
Section 11.2, Error Codes When Using H.323
Section 11.5, Failure Response Codes When Using SIP, under the subheading "SIP Message Error Codes"

3.4.11 Dialogic® Global Call ISDN Technology Guide

There are currently no updates to this document.

3.4.12 Dialogic® Global Call SS7 Technology Guide

Add that the Global Call SS7 binaries are now linked with the dynamic link library in the Dialogic® SS7 DSI Development Package. (IPY00081381)

This change requires that Global Call SS7 customers use the Dialogic® SS7 DSI Development Package version 5.0 or later. If a customer is using an earlier version, the Global Call SS7 server will not start during download.

Note: The Dialogic® SS7 DSI Development Package installation does not put the *gctlib.dll* file in an accessible location to be dynamically linked with *Dlgcs7srv.exe*. For this to happen, you need to copy the *gctlib.dll* file to the system32 directory or add the SS7 SDK installation directory (usually the C:\Septel directory) to the PATH environment variable.

For addtional information about SS7 products, refer to the *Dialogic® Global Call SS7 Technology Guide* and the *Dialogic® DSI SS7HD Network Interface Boards Programmer's Manual Issue 10*.

3.4.13 Dialogic® IP Media Library API Programming Guide

Update to **Chapter 8, Quality of Service (QoS) Alarms**

With the Service Update, two new QoS alarms are supported on Dialogic® IPT Boards:

- QOSTYPE_RTCPTIMEOUT - QoS alarm for Real Time Control Protocol (RTCP) inactivity
- QOSTYPE_RTPTIMEOUT - QoS alarm for Real Time Protocol (RTP) inactivity

For further information about these alarms, see Section 1.19, "New QoS Alarms for RTCP and RTP Inactivity", on page 61 of this Release Update.

Update to **Section 8.8, Network Failure Alarm**

Because of a new feature in the Service Update, this section should be updated to indicate that it is applicable to Dialogic® DM/IP601-CPCI-100BT and Dialogic® DM/IP601-2E1-CPCI-100BT IP Boards, as well as to Dialogic® IPT Boards. For further information, see Section 1.8, "Media LAN Disconnection Alarm Notification for Dialogic® DM/IP Boards", on page 33 of this Release Update.

3.4.14 Dialogic® IP Media Library API Library Reference

Updates for new QoS alarms

With the Service Update, two new Quality of Service (QoS) alarms are supported on Dialogic® IPT Boards:

- QOSTYPE_RTCPTIMEOUT - QoS alarm for Real Time Control Protocol (RTCP) inactivity

- QOSTYPE_RTPTIMEOUT - QoS alarm for Real Time Protocol (RTP) inactivity

These alarms and their associated events should be included in the following function and data structure reference pages: **ipm_DisableEvents( )**, **ipm_EnableEvents( )**, IPM_QOS_ALARM_DATA, IPM_QOS_SESSION_INFO, IPM_QOS_THRESHOLD_DATA.

For further information about these alarms, see Section 1.19, "New QoS Alarms for RTCP and RTP Inactivity", on page 61 of this Release Update.

Update to **Chapter 2, Function Information**

A new API, **ipm_ResetChannel( ),** has been added. See Section 1.29, "Media Channel Reset Capability (Stuck IP Media Channel Recovery)", on page 78 of this Release Update for more details.

Update to **ipm_DisableEvents( )** and **ipm_EnableEvents( )**

Because of a new feature in the Service Update, the EVT_NETWORKFAILURE event is now supported on Dialogic® DM/IP601-CPCI-100BT and Dialogic® DM/IP601-2E1-CPCI-100BT IP Boards, as well as on Dialogic® IPT Boards. For further information, see Section 1.8, "Media LAN Disconnection Alarm Notification for Dialogic® DM/IP Boards", on page 33 of this Release Update.

Updates to **ipm_ModifyMedia( )**

In the description of the **eDirection** parameter, DATA_MULTICAST_SERVER should be deleted. The session **eDirection** setting cannot be modified once it is set.

The names of the termination events given for the **ipm_ModifyMedia( )** function, IPMEV_MODIFY_MEDIA and IPMEV_MODIFY_MEDIA_FAIL, are incorrect. The correct event names are IPMEV_MODIFYMEDIA and IPMEV_MODIFYMEDIA_FAIL.

Update to **ipm_SetRemoteMediaInfo( )**

Because of a new feature in the Service Update, there is a new value for the **eDirection** parameter:

- DATA_MULTICAST_CLIENT – multicast client mode (supported for Dialogic® DM/IP Boards only)

For information about this feature, see Section 1.15, "IP Multicast Client Support", on page 55 of this Release Update.

Update to **ipm_StartMedia( )**

Because of a new feature in the Service Update, there is a new value for the **eDirection** parameter:

- DATA_MULTICAST_CLIENT – multicast client mode (supported for Dialogic® DM/IP Boards only)

For information about this feature, see Section 1.15, "IP Multicast Client Support", on page 55 of this Release Update.

Update to **Chapter 3, Events**

The names of the termination events given for the **ipm_ModifyMedia( )** function, IPMEV_MODIFY_MEDIA and IPMEV_MODIFY_MEDIA_FAIL, are incorrect. The correct event names are IPMEV_MODIFYMEDIA and IPMEV_MODIFYMEDIA_FAIL.

Updates to **IPM_QOS_ALARM_DATA**

Because of a new feature in the Service Update, the eQoSType value QOSTYPE_NETWORKFAILURE is now supported on Dialogic® DM/IP601-CPCI-100BT and Dialogic® DM/IP601-2E1-CPCI-100BT IP Boards, as well as on Dialogic®

IPT Boards. For further information, see Section 1.8, “Media LAN Disconnection Alarm Notification for Dialogic® DM/IP Boards”, on page 33 of this Release Update.

A new eQoSType value has been added to the IPM_QOS_ALARM_DATA structure. See Section 1.25, “Modified Alarm Events for Media LAN Disconnect”, on page 74 of this Release Update for more details.

3.4.15 Dialogic® Modular Station Interface API Programming Guide

There are currently no updates to this document.

3.4.16 Dialogic® Modular Station Interface API Library Reference

There are currently no updates to this document.

3.4.17 Dialogic® Standard Runtime Library API Programming Guide

There are currently no updates to this document.

3.4.18 Dialogic® Standard Runtime Library API Library Reference

There are currently no updates to this document.

3.4.19 Dialogic® Voice API Programming Guide

Functions not supported

The **r2_creatfsig( )** and **r2_playbsig( )** functions, which were previously provided for backward compatibility only, are no longer supported. All references to these functions should be deleted. R2MF signaling is typically accomplished through the Dialogic® Global Call API.

Update to **Chapter 6, Application Development Guidelines**

The following note should be added to **Section 6.4.2, Multithreading and Multiprocessing**:

Note: The continuous speech processing architecture allows a voice channel to be shared between processes (or applications) on Dialogic® JCT Boards, on Dialogic® DM3 Boards, or on Dialogic® Host Media Processing (HMP) (starting with Dialogic® Host Media Processing Software Release 1.3 for Windows®), providing one process does the play activity and the other process does the record/stream activity. Other CSP scenarios are **not** supported, such as playing or recording/streaming from both processes. For details, refer to the application note, Telephony Application Architectures for Dialogic® Boards with DM3 Architecture, located at http://www.dialogic.com/products/tdm_boards/media_processing/docs/9380an.pdf.

Speed control on Dialogic® DM3 Boards using 6 kHz coders

The following coders are now supported and should be added to **Section 9.1**, **Speed and Volume Control Overview**:

- OKI ADPCM 24 kbps (6 kHz 4-bit)
- G.711 PCM A-law 48 kbps (6 kHz 8-bit)
- G.711 PCM mu-law 48 kbps (6 kHz 8-bit)

3.4.20 Dialogic® Voice API Library Reference

Functions not supported

The **r2_creatfsig( )** and **r2_playbsig( )** functions, which were previously provided for backward compatibility only, are no longer supported. All references to these functions should be deleted. R2MF signaling is typically accomplished through the Dialogic® Global Call API.

Update to **dx_getdig( )** (IPY00038453)

For Dialogic® DM3 Boards, the return value of **dx_getdig( )** in synchronous mode has been changed to return 0 instead of 1 when there are no digits in the buffer. The NULL character in the digit string 'dg_value' is no longer counted as a digit. Similarly, when **dx_getdig( )** returns the number of digits, the terminating NULL is no longer added to the number of digits. (The NULL was previously counted in the numdig return value calculation, but since it is not a digit, the NULL is no longer included.)

For Dialogic® Springware Boards, the terminating NULL **is** included in the number of digits. So for Springware Boards, **dx_getdig( )** still returns 1 when there are no digits in the buffer.

Update to **dx_OpenStreamBuffer( )** (IPY00044981)

The following caution should be added for **dx_OpenStreamBuffer( )**:

- When using Dialogic® DM3 Boards, the **dx_open( )** function must be called on a board, channel, or physical board before **dx_OpenStreamBuffer( )** is called. Failure to do so would prevent the DM3 library from loading, and **dx_OpenStreamBuffer( )** would fail.

Update to **dx_rec( )**, **dx_reciottdata( )**, **dx_recvox( )**, and **dx_recwav( )**

The **dx_rec( )**, **dx_reciottdata( )**, **dx_recvox( )**, and **dx_recwav( )** function reference pages contain a caution about channels getting stuck when failing to listen to a TDM bus time slot prior to invoking a record operation. The caution should be revised: this condition now returns an error rather than resulting in a stuck channel. The revised caution is:

- On Dialogic® DM3 Boards using a flexible routing configuration, voice channels must be listening to a TDM bus time slot in order for voice recording functions, such as **dx_reciottdata( )** and others, to work. The actual recording operation will start only after the voice channel is listening to the proper external time slot. In other words, you must issue a **dx_listen( )** function call on the device handle before calling a voice recording function for that device handle, and the **dx_listen( )** has to be called from the same process as the voice recording function. If not, the voice recording function will return TDX_ERROR with EDX_SH_MISSING as the termination reason.

Update to **dx_setevtmsk( )** (IPY00038053)

The following information should be added to the description of the mask parameter:

User defined tones that are associated an optional digit (**dx_addtone( )**) have digit reporting enabled by default in Dialogic® System Release 6.1 CompactPCI for Windows®. The user defined tones digit reporting can be turned off by using **dx_setevtmsk( )** with DM_DIGOFF mask. To reactivate digit reporting, use **dx_setevtmsk( )** with DM_DIGITS mask.

3.5 Demonstration Software Documentation Updates

This section contains updates to the following documents (click the title to jump to the corresponding section):

- Dialogic® Continuous Speech Processing API Demo Guide
- Dialogic® Global Call API Demo Guide
- Dialogic® High Availability for Windows® Demo Guide
- Dialogic® IP Gateway (Global Call) Demo Guide
- Dialogic® IP Media Server (Global Call) Demo Guide
- Dialogic® IP Media Gateway (IPML) Demo Guide

3.5.1 Dialogic® Continuous Speech Processing API Demo Guide

There are currently no updates to this document.

3.5.2 Dialogic® Global Call API Demo Guide

There are currently no updates to this document.

3.5.3 Dialogic® High Availability for Windows® Demo Guide

There are currently no updates to this document.

3.5.4 Dialogic® IP Gateway (Global Call) Demo Guide

There are currently no updates to this document.

3.5.5 Dialogic® IP Media Server (Global Call) Demo Guide

Chapter 1, Demo Description, should contain the following note (IPY00006586 = PTR# 36565):

Note: When used with a Dialogic® DM/IP or Dialogic® IPT Board, the IP Media Server demo does not support fax or conference capabilities even though those functions appear in some of the demo's menus.

Section 4.3.2, Using the Media Server, contains a list of menu options in the section headed “Main Menu [Main_Menu]” which should be corrected as follows:

The menu includes a fifth option which is not included in the list. However, the missing option (2 - Fax) is not supported when using a Dialogic® DM/IP or Dialogic® IPT Board.

The listing for the “3 - Conferencing” menu option includes a parenthetical comment which refers to the Dialogic® Host Media Processing (HMP) product and is therefore irrelevant for Dialogic® System Releases. The item should include a note that indicates that this option is not supported when using a Dialogic® DM/IP or Dialogic® IPT Board.

3.5.6 Dialogic® IP Media Gateway (IPML) Demo Guide

There are currently no updates to this document.