



# **Dialogic® NaturalAccess™ NaturalFax™ API Developer's Manual**

# Copyright and legal notices

---

Copyright © 2000-2010 Dialogic Corporation. All Rights Reserved. You may not reproduce this document in whole or in part without permission in writing from Dialogic Corporation at the address provided below.

All contents of this document are furnished for informational use only and are subject to change without notice and do not represent a commitment on the part of Dialogic Corporation or its subsidiaries ("Dialogic"). Reasonable effort is made to ensure the accuracy of the information contained in the document. However, Dialogic does not warrant the accuracy of this information and cannot accept responsibility for errors, inaccuracies or omissions that may be contained in this document.

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH DIALOGIC® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN A SIGNED AGREEMENT BETWEEN YOU AND DIALOGIC, DIALOGIC ASSUMES NO LIABILITY WHATSOEVER, AND DIALOGIC DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF DIALOGIC PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY INTELLECTUAL PROPERTY RIGHT OF A THIRD PARTY.

Dialogic products are not intended for use in medical, life saving, life sustaining, critical control or safety systems, or in nuclear facility applications.

Due to differing national regulations and approval requirements, certain Dialogic products may be suitable for use only in specific countries, and thus may not function properly in other countries. You are responsible for ensuring that your use of such products occurs only in the countries where such use is suitable. For information on specific products, contact Dialogic Corporation at the address indicated below or on the web at [www.dialogic.com](http://www.dialogic.com).

It is possible that the use or implementation of any one of the concepts, applications, or ideas described in this document, in marketing collateral produced by or on web pages maintained by Dialogic may infringe one or more patents or other intellectual property rights owned by third parties. Dialogic does not provide any intellectual property licenses with the sale of Dialogic products other than a license to use such product in accordance with intellectual property owned or validly licensed by Dialogic and no such licenses are provided except pursuant to a signed agreement with Dialogic. More detailed information about such intellectual property is available from Dialogic's legal department at 9800 Cavendish Blvd., 5th Floor, Montreal, Quebec, Canada H4M 2V9. Dialogic encourages all users of its products to procure all necessary intellectual property licenses required to implement any concepts or applications and does not condone or encourage any intellectual property infringement and disclaims any responsibility related thereto. These intellectual property licenses may differ from country to country and it is the responsibility of those who develop the concepts or applications to be aware of and comply with different national license requirements.

Any use case(s) shown and/or described herein represent one or more examples of the various ways, scenarios or environments in which Dialogic® products can be used. Such use case(s) are non-limiting and do not represent recommendations of Dialogic as to whether or how to use Dialogic products.

Dialogic, Dialogic Pro, Brooktrout, Diva, Cantata, SnowShore, Eicon, Eicon Networks, NMS Communications, NMS (stylized), Eiconcard, SIPcontrol, Diva ISDN, TruFax, Exnet, EXS, SwitchKit, N20, Making Innovation Thrive, Connecting to Growth, Video is the New Voice, Fusion, Vision, PacketMedia, NaturalAccess, NaturalCallControl, NaturalConference, NaturalFax and Shiva, among others as well as related logos, are either registered trademarks or trademarks of Dialogic Corporation or its subsidiaries. Dialogic's trademarks may be used publicly only with permission from Dialogic. Such permission may only be granted by Dialogic's legal department at 9800 Cavendish Blvd., 5th Floor, Montreal, Quebec, Canada H4M 2V9. Any authorized use of Dialogic's trademarks will be subject to full respect of the trademark guidelines published by Dialogic from time to time and any use of Dialogic's trademarks requires proper acknowledgement.

Windows is a registered trademark of Microsoft Corporation in the United States and/or other countries. The names of actual companies and product mentioned herein are the trademarks of their respective owners.

This document discusses one or more open source products, systems and/or releases. Dialogic is not responsible for your decision to use open source in connection with Dialogic products (including without limitation those referred to herein), nor is Dialogic responsible for any present or future effects such usage might have, including without limitation effects on your products, your business, or your intellectual property rights.

## Revision history

---

| Revision                  | Release date   | Notes                                |
|---------------------------|----------------|--------------------------------------|
| 9000-60009-10             | July 2000      | LBG, Platform Support for Fusion 4.0 |
| 9000-60009-11             | September 2000 | LBG, NaturalFax 3.4                  |
| 9000-60009-12             | September 2000 | LBG, NaturalFax 4.0 beta             |
| 9000-60009-13             | January 2001   | MCM, NaturalFax 4.0                  |
| 9000-60009-14             | May 2001       | MCM, NaturalFax 4.1 beta             |
| 9000-60009-15             | August 2001    | MCM, NACD 2001-1                     |
| 9000-60009-16             | November 2001  | MCM, NACD 2002-1 Beta                |
| 9000-60009-17             | May 2002       | LBG, NACD 2002-1                     |
| 9000-60009-18             | April 2003     | LBG, Natural Access 2003-1           |
| 9000-60009-19             | December, 2003 | LBG, Natural Access 2004-1           |
| 9000-60009-20             | March, 2005    | LBG, Natural Access 2005-1           |
| 64-0513-01                | October, 2009  | LBG, NaturalAccess R9.0              |
| 64-0513-02                | December, 2009 | LBG, NaturalAccess R9.0.1            |
| 64-0513-03 Rev A          | October, 2010  | LBG, NaturalAccess R9.0.4            |
| Last modified: 2010-10-15 |                |                                      |

Refer to [www.dialogic.com](http://www.dialogic.com) for product updates and for information about support policies, warranty information, and service offerings.

# Table Of Contents

---

|   |           |
|---|-----------|
| <b>1. Introduction.....</b>                           | <b>7</b>  |
| <b>2. Overview of the NaturalFax service .....</b>    | <b>10</b> |
| NaturalFax definition.....                            | 10        |
| Supported Features .....                              | 10        |
| NaturalFax and NMS hardware .....                     | 10        |
| Fax application overview .....                        | 10        |
| Natural Access components .....                       | 11        |
| Preparing the hardware environment .....              | 12        |
| <b>3. Configuring NaturalFax .....</b>                | <b>13</b> |
| Overview of configuring NaturalFax .....              | 13        |
| Setting environment variables in Windows.....         | 13        |
| Setting environment variables in UNIX.....            | 13        |
| Modifying the Natural Access configuration file ..... | 14        |
| NaturalFax services.....                              | 15        |
| Sample Natural Access configuration file.....         | 15        |
| Modifying the board keyword file .....                | 17        |
| Configuring AG boards .....                           | 19        |
| Configuring CG boards .....                           | 19        |
| Sample board keyword files .....                      | 20        |
| Sample AG 2000/200 board keyword file .....           | 20        |
| Sample CG 6565 board keyword file.....                | 20        |
| Sample CG 6060 board keyword file.....                | 22        |
| Verifying NaturalFax .....                            | 23        |
| Verifying the NaturalFax library .....                | 23        |
| Sending and receiving a fax .....                     | 23        |
| <b>4. Developing applications .....</b>               | <b>24</b> |
| A typical fax application .....                       | 24        |
| A typical fax and voice application .....             | 25        |
| Setting up the Natural Access environment.....        | 25        |
| Initializing NaturalFax .....                         | 26        |
| Creating event queues and contexts .....              | 27        |
| Opening services .....                                | 27        |
| Establishing a call .....                             | 29        |
| Placing a call.....                                   | 29        |
| Receiving a call .....                                | 30        |
| Using document queues .....                           | 30        |
| Building a document queue.....                        | 30        |
| Transmitting and receiving faxes .....                | 31        |
| Transmitting faxes.....                               | 31        |
| Receiving faxes .....                                 | 31        |
| Polling the called fax terminal .....                 | 31        |
| Answering a poll request .....                        | 32        |
| Resetting a document queue.....                       | 33        |
| Performing offline image conversion .....             | 33        |
| Performing online image conversion.....               | 34        |
| Image conversion during fax transmission .....        | 35        |
| Image conversion during fax reception.....            | 38        |
| Generating TIFF-S files on receive .....              | 39        |
| Monitoring fax session status .....                   | 40        |

|   |           |
|---|-----------|
| Tracing NaturalFax applications .....                     | 40        |
| Error handling during a fax session .....                 | 41        |
| Terminating and shutting down .....                       | 41        |
| Tearing down or resetting a document queue .....          | 42        |
| Closing Natural Access services .....                     | 43        |
| <b>5. Optimizing performance.....</b>                     | <b>44</b> |
| Strategies for optimizing performance.....                | 44        |
| Maximizing ports for fax transmission .....               | 45        |
| Maximizing universal ports .....                          | 45        |
| <b>6. Working with images .....</b>                       | <b>47</b> |
| Image format characteristics .....                        | 47        |
| Encoding formats.....                                     | 47        |
| Resolution formats.....                                   | 49        |
| Page width formats .....                                  | 49        |
| Options for storing and converting image data .....       | 50        |
| T.37 and TIFF-S .....                                     | 51        |
| <b>7. Function summary .....</b>                          | <b>52</b> |
| Document queue functions .....                            | 52        |
| Transmit and receive documents functions .....            | 52        |
| Image format conversion functions.....                    | 53        |
| Managing pages and document contents functions.....       | 53        |
| Status monitoring functions .....                         | 53        |
| <b>8. Function reference .....</b>                        | <b>54</b> |
| Using the function reference .....                        | 54        |
| nfxAnswerFaxPoll .....                                    | 55        |
| nfxCheckTIFF .....  | 58        |
| nfxConvertFileDirect .....                                | 61        |
| nfxCreateQueue .....                                      | 62        |
| nfxDestroyQueue .....                                     | 63        |
| nfxEnqueueDoc .....                                       | 65        |
| nfxGetDocStatus .....                                     | 66        |
| nfxGetSessionStatus.....                                  | 68        |
| nfxMergeFile .....  | 69        |
| nfxReceiveFax.....  | 71        |
| nfxResetQueue.....  | 75        |
| nfxSendFax .....  | 75        |
| nfxSplitFile .....  | 81        |
| nfxStopSession .....                                      | 82        |
| <b>9. Demonstration programs and utilities.....</b>       | <b>84</b> |
| Summary of the demonstration programs and utilities ..... | 84        |
| Using voice and fax: caller.....                          | 84        |
| Using voice and fax: faxback .....                        | 86        |
| Receiving a fax: nfxrecv .....                            | 88        |
| Sending a fax: nfxsend .....                              | 90        |
| Verifying TIFF-F or TIFF-S format: nfxcheck.....          | 92        |
| Converting TIFF-F or TIFF-S files: nfxcnvrt .....         | 92        |
| Merging TIFF-F or TIFF-S files: nfxmerge .....            | 93        |
| Splitting TIFF-F or TIFF-S files: nfxsplit .....          | 93        |
| Converting ASCII files: nfxtxttf .....                    | 94        |
| <b>10. Errors.....</b>                                    | <b>95</b> |

|   |            |
|---|------------|
| Alphabetical error summary .....                        | 95         |
| Numerical error summary .....                           | 97         |
| <b>11. Events .....</b>                                 | <b>99</b>  |
| Overview of events .....                                | 99         |
| Completion events .....                                 | 100        |
| NFXEVN_SESSION_DONE values .....                        | 100        |
| Informational events .....                              | 104        |
| NFXEVN_DOC_END values .....                             | 105        |
| NFXEVN_DOC_END and NFXEVN_PAGE_END values .....         | 105        |
| Confirmation events .....                               | 107        |
| <b>12. Data structures .....</b>                        | <b>109</b> |
| Overview of NaturalFax data structures .....            | 109        |
| NFX_TRANSMIT_PARMS .....                                | 110        |
| NFX_RECEIVE_PARMS .....                                 | 115        |
| NFX_DOC_PARMS .....                                     | 119        |
| NFX_CONVERT_PARMS .....                                 | 120        |
| Status structures .....                                 | 122        |
| <b>13. DSP requirements for AG and CG boards .....</b>  | <b>123</b> |
| AG board DSP requirements for NaturalFax .....          | 123        |
| CG board DSP requirements for NaturalFax .....          | 123        |
| <b>14. Group 3 fax technology .....</b>                 | <b>124</b> |
| Fax protocols .....                                     | 124        |
| Group 3 protocol .....                                  | 124        |
| Phase A - Call setup .....                              | 125        |
| Phase B - Pre-message procedure .....                   | 125        |
| Phase C - Image transfer and message transmission ..... | 126        |
| Phase D - Post-message procedures .....                 | 126        |
| Phase E - Call release .....                            | 127        |
| Non-standard facilities frame .....                     | 127        |
| <b>15. Modem metrics .....</b>                          | <b>128</b> |
| Overview of modem metrics .....                         | 128        |
| Signal to noise ratio (SNR) .....                       | 128        |
| Training check frame (TCF) .....                        | 129        |
| Using SNR and TCF together .....                        | 130        |
| <b>16. Sub-addressing .....</b>                         | <b>131</b> |
| Overview of sub-addressing .....                        | 131        |
| Using sub-addressing .....                              | 132        |
| <b>17. Index .....</b>                                  | <b>133</b> |

# 1. Introduction

---

The *Dialogic® NaturalAccess™ NaturalFax™ API Developer's Manual* provides:

- Detailed information about developing NaturalFax applications under NaturalAccess.
- A complete reference for the NaturalFax API.
- Information on the NaturalFax demonstration programs and utilities.

This manual is targeted to developers of fax applications who are using NaturalAccess. This document defines telephony terms where applicable, but assumes that you are familiar with telephony concepts and the C programming language.

## Terminology

---

**Note:** The product to which this document pertains is part of the NMS Communications Platforms business that was sold by NMS Communications Corporation ("NMS") to Dialogic Corporation ("Dialogic") on December 8, 2008. Accordingly, certain terminology relating to the product has been changed. Below is a table indicating both terminology that was formerly associated with the product, as well as the new terminology by which the product is now known. This document is being published during a transition period; therefore, it may be that some of the former terminology will appear within the document, in which case the former terminology should be equated to the new terminology, and vice versa.

| Former terminology | Dialogic terminology                                  |
|--------------------|---|
| CG 6060 Board      | Dialogic® CG 6060 PCI Media Board                     |
| CG 6060C Board     | Dialogic® CG 6060C CompactPCI Media Board             |
| CG 6565 Board      | Dialogic® CG 6565 PCI Media Board                     |
| CG 6565C Board     | Dialogic® CG 6565C CompactPCI Media Board             |
| CG 6565e Board     | Dialogic® CG 6565E PCI Express Media Board            |
| CX 2000 Board      | Dialogic® CX 2000 PCI Station Interface Board         |
| CX 2000C Board     | Dialogic® CX 2000C CompactPCI Station Interface Board |
| AG 2000 Board      | Dialogic® AG 2000 PCI Media Board                     |
| AG 2000C Board     | Dialogic® AG 2000C CompactPCI Media Board             |
| AG 2000-BRI Board  | Dialogic® AG 2000-BRI Media Board                     |
| NMS OAM Service    | Dialogic® NaturalAccess™ OAM API                      |

| Former terminology                | Dialogic terminology  |
|-----------------------------------|---|
| NMS OAM System                    | Dialogic® NaturalAccess™ OAM System                           |
| NMS SNMP                          | Dialogic® NaturalAccess™ SNMP API                             |
| Natural Access                    | Dialogic® NaturalAccess™ Software                             |
| Natural Access Service            | Dialogic® NaturalAccess™ Service                              |
| Fusion                            | Dialogic® NaturalAccess™ Fusion™ VoIP API                     |
| ADI Service                       | Dialogic® NaturalAccess™ Alliance Device Interface API        |
| CDI Service                       | Dialogic® NaturalAccess™ CX Device Interface API              |
| Digital Trunk Monitor Service     | Dialogic® NaturalAccess™ Digital Trunk Monitoring API         |
| MSPP Service                      | Dialogic® NaturalAccess™ Media Stream Protocol Processing API |
| Natural Call Control Service      | Dialogic® NaturalAccess™ NaturalCallControl™ API              |
| NMS GR303 and V5 Libraries        | Dialogic® NaturalAccess™ GR303 and V5 Libraries               |
| Point-to-Point Switching Service  | Dialogic® NaturalAccess™ Point-to-Point Switching API         |
| Switching Service                 | Dialogic® NaturalAccess™ Switching Interface API              |
| Voice Message Service             | Dialogic® NaturalAccess™ Voice Control Element API            |
| NMS CAS for Natural Call Control  | Dialogic® NaturalAccess™ CAS API                              |
| NMS ISDN                          | Dialogic® NaturalAccess™ ISDN API                             |
| NMS ISDN for Natural Call Control | Dialogic® NaturalAccess™ ISDN API                             |
| NMS ISDN Messaging API            | Dialogic® NaturalAccess™ ISDN Messaging API                   |
| NMS ISDN Supplementary Services   | Dialogic® NaturalAccess™ ISDN API Supplementary Services      |
| NMS ISDN Management API           | Dialogic® NaturalAccess™ ISDN Management API                  |



| Former terminology                           | Dialogic terminology   |
|--|--|
| NaturalConference Service                    | Dialogic® NaturalAccess™ NaturalConference™ API  |
| NaturalFax                                   | Dialogic® NaturalAccess™ NaturalFax™ API   |
| SAI Service                                  | Dialogic® NaturalAccess™ Universal Speech Access API                                       |
| NMS SIP for Natural Call Control             | Dialogic® NaturalAccess™ SIP API   |
| NMS RJ-45 interface                          | Dialogic® MD1 RJ-45 interface  |
| NMS RJ-21 interface                          | Dialogic® MD1 RJ-21 interface  |
| NMS Mini RJ-21 interface                     | Dialogic® MD1 Mini RJ-21 interface   |
| NMS Mini RJ-21 to NMS RJ-21 cable            | Dialogic® MD1 Mini RJ-21 to MD1 RJ-21 cable  |
| NMS RJ-45 to two 75 ohm BNC splitter cable   | Dialogic® MD1 RJ-45 to two 75 ohm BNC splitter cable                                       |
| NMS signal entry panel                       | Dialogic® Signal Entry Panel   |
| Video Access Utilities                       | Dialogic® NaturalAccess™ Video Access Toolkit Utilities                                    |
| Video Mail Application Demonstration Program | Dialogic® NaturalAccess™ Video Access Toolkit Video Mail Application Demonstration Program |
| Video Messaging Server Interface             | Dialogic® NaturalAccess™ Video Access Toolkit Video Messaging Server Interface             |
| 3G-324M Interface                            | Dialogic® NaturalAccess™ Video Access Toolkit 3G-324M Interface                            |

## 2. Overview of the NaturalFax service

---

### NaturalFax definition

NaturalFax is a C function library component of Natural Access that provides fax functionality. Use NaturalFax to produce fax server products for enterprise fax applications, IP telephony gateways, and large scale fax service provider systems. Applications include LAN fax, fax broadcast, fax-on-demand, store-and-forward, and realtime fax-over-IP data networks. NaturalFax applications can transmit and receive Group 3 facsimiles at rates of up to 14,400 b/s.

### Supported Features

NaturalFax supports the following features:

- Enhanced Group 3 fax features such as:
  - 1D, 2D, and MMR encoding
  - Error correction mode (ECM)
  - Low, high, and superhigh resolutions
  - Standard page width formats
- The T.37 format, specified as 1D encoding, low resolution, and A4 page width
- The ability to convert documents into different formats, resolutions, or encodings online (on-the-fly) or offline
- The ability to poll a remote fax terminal and request that it transmit a fax.

For more information, refer to Fax protocols and Group 3 protocol.

### NaturalFax and NMS hardware

NaturalFax runs on NMS hardware. This hardware consists of various telecommunications interfaces, high-speed controllers, and arrays of DSPs.

On AG and CG boards, the fax modems run on TI C5x and C54x DSPs, sharing them with other telephony algorithms such as voice record and play. The T.30 fax communication protocol runs on the AG or CG board controller and host processor. This sharing of DSP and controller resources contrasts with traditional chip-per-port designs, and provides the highest density and lowest cost-per-port fax boards and systems.

### Fax application overview

A NaturalFax application uses Natural Access functions to control the PSTN and NaturalFax functions to manage fax sessions and related tasks. The Natural Call Control service is responsible for establishing and disconnecting the calls.

Natural Access functions:

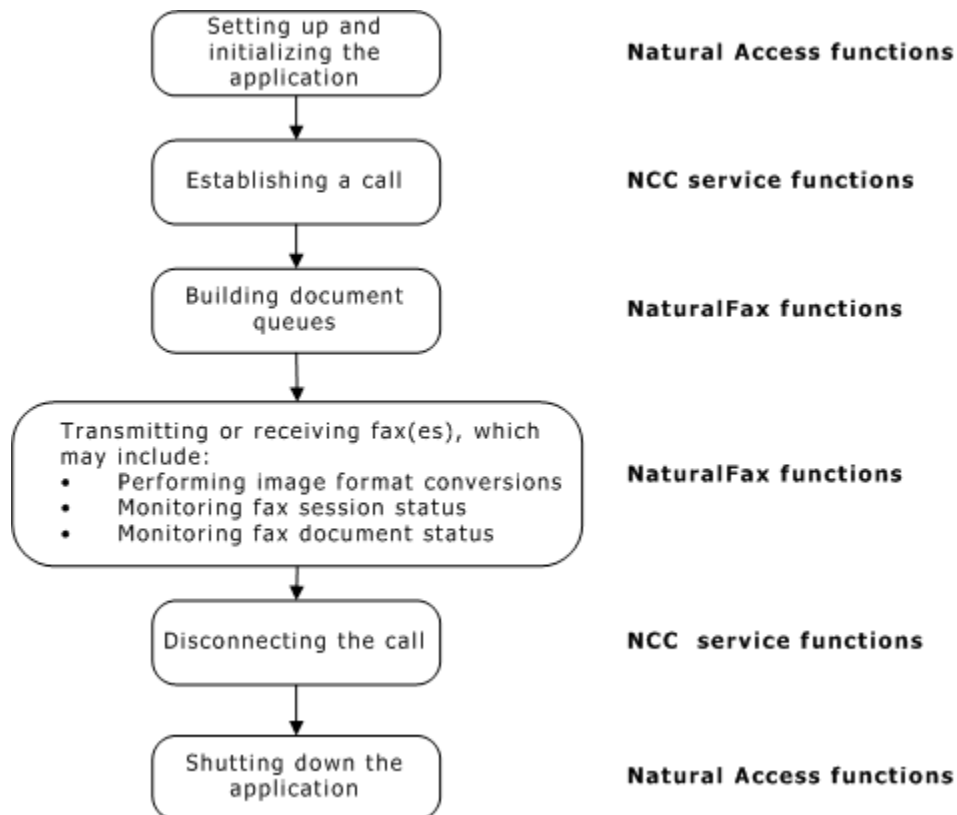
- Initialize and tear down the application.
- Perform call control.

NaturalFax functions:

- Build and destroy document queues.
- Transmit or receive faxes.
- Perform image format conversions.
- Monitor fax session status.
- Monitor fax document status.

A fax session begins when a transmit or receive operation is initiated, and ends when the application receives the corresponding NaturalFax DONE event. A document queue contains a list of one or more document files (the document files must be in TIFF-F or TIFF-S format). NaturalFax functions operate on document queues, not on individual files. You must use document queues for all fax operations, including transmitting and receiving single documents.

The following illustration shows how Natural Access and NaturalFax are used in a basic NaturalFax application:



## Natural Access components

A Natural Access service is a group of logically related telephony functions. NaturalFax is a Natural Access service that provides functions for sending and receiving Group 3 faxes.

Within Natural Access, services and accompanying resources are organized around a single processing entity called a context. A context usually represents an application instance controlling a single telephone call. Some Natural Access contexts are not associated with a call. For example, an operation performing fax file conversions does not require a telephone line.

**Note:** Only one asynchronous fax operation per context can be active at one time.

A service can only be opened once on an individual context. For example, in order to send faxes using all eight channels of an AG 2000 board, an application needs to create eight Natural Access contexts, and open the NaturalFax service on each context.

An event queue is the communication path from a service to an application (such as NaturalFax). A service generates events indicating certain conditions or state changes. An application retrieves the events from the event queue.

You can alter the characteristics of services by modifying associated parameters. Each NaturalFax parameter structure has default values that are sufficient for most configurations. NaturalFax applications typically need to set the subscriber ID (the telephone number) to comply with FCC or other local regulations, but may not need to modify other parameters.

Natural Access manages parameters for services on a context-by-context basis. The context maintains a copy of the parameters for all services opened on the context, allowing each fax operation to have its own characteristics. Refer to the Overview of NaturalFax data structures for detailed descriptions of NaturalFax parameters and their default values.

For detailed information about Natural Access, refer to the *Natural Access Developer's Reference Manual*.

## Preparing the hardware environment

You must initialize and load all boards before you can run NaturalFax. Use NMS OAM to perform hardware configuration and initialization. For more information, refer to the *NMS OAM System User's Manual*.

## 3. Configuring NaturalFax

### Overview of configuring NaturalFax

Configure NaturalFax for your system before using it to develop applications. Complete the following steps to configure NaturalFax:

| Step | Action   |
|------|--|
| 1    | Set the environment variables for your platform.   |
| 2    | Modify the Natural Access configuration file to define all services that the application will use.   |
| 3    | Modify the board keyword files to allocate the DSPs for fax functions and to specify the NaturalFax runtime software to download when the board initializes. |
| 4    | Verify that NaturalFax is properly installed and configured.   |

### Setting environment variables in Windows

During software installation, the installation program modifies the following environment variables (unless you specify not to):

```
NFXTEXTFONT=c:\ nms\nfx\text.fnt  
NFXHEADERFONT=c:\ nms\nfx\header.fnt
```

If you changed the default directory during installation, replace `c:\ nms` with the path you specified for your system.

If you specified that the installation should not modify the environment variables, modify the variables using the Windows Control Panel before running any NaturalFax applications.

Refer to [Modifying the Natural Access configuration file](#) for information on completing the installation.

### Setting environment variables in UNIX

NaturalFax can run from any user environment, including the superuser environment, if the environment variables NFXTEXTFONT and NFXHEADERFONT are properly configured for that user. If these environment variables were not set up during the NaturalFax installation, make sure they are added to the appropriate user environment.

Complete the following steps to set the NaturalFax environment variables:

| Step | Action  |
|------|---|
| 1    | Open a terminal window.   |
| 2    | Log on as the designated user for the NaturalFax application. If you will be running the NaturalFax application from the root user environment, log in as root at your shell's command prompt by entering:<br><pre>su -</pre> |

| Step              | Action   |   |                   |                                     |                |                  |   |   |                |   |
|-------------------|--|---|-------------------|-------------------------------------|----------------|------------------|---|---|----------------|---|
| 3                 | Enter the password when prompted.  |   |                   |                                     |                |                  |   |   |                |   |
| 4                 | <div>With your text editor, open the shell script file executed by the user's logon shell as part of the logon sequence and add the following environment variables:</div> <table><tr><th>For this shell...</th><th>Edit this file...</th><th>Create this environment variable...</th></tr><tr><td>Bourne or Korn</td><td><i>/.profile</i></td><td><div>Create the environment variable NFXTEXTFONT to specify the text font file. For example:</div><div>NFXTEXTFONT=/opt/nms/nfx/text.fnt<br/>export NFXTEXTFONT</div><div>Create the environment variable NFXHEADERFONT to specify the header font file. For example:</div><div>NFXHEADERFONT=/opt/nms/nfx/header.fnt<br/>export NFXHEADERFONT</div></td></tr><tr><td>C</td><td><i>/.login</i></td><td><div>Create the environment variable NFXTEXTFONT to specify the text font file. For example:</div><div>setenv NFXTEXTFONT /opt/nms/nfx/text.fnt</div><div>Create the environment variable NFXHEADERFONT to specify the header font file. For example:</div><div>setenv NFXHEADERFONT /opt/nms/nfx/header.fnt</div></td></tr></table> | For this shell...   | Edit this file... | Create this environment variable... | Bourne or Korn | <i>/.profile</i> | <div>Create the environment variable NFXTEXTFONT to specify the text font file. For example:</div> <div>NFXTEXTFONT=/opt/nms/nfx/text.fnt<br/>export NFXTEXTFONT</div> <div>Create the environment variable NFXHEADERFONT to specify the header font file. For example:</div> <div>NFXHEADERFONT=/opt/nms/nfx/header.fnt<br/>export NFXHEADERFONT</div> | C | <i>/.login</i> | <div>Create the environment variable NFXTEXTFONT to specify the text font file. For example:</div> <div>setenv NFXTEXTFONT /opt/nms/nfx/text.fnt</div> <div>Create the environment variable NFXHEADERFONT to specify the header font file. For example:</div> <div>setenv NFXHEADERFONT /opt/nms/nfx/header.fnt</div> |
| For this shell... | Edit this file...  | Create this environment variable...   |                   |                                     |                |                  |   |   |                |   |
| Bourne or Korn    | <i>/.profile</i>   | <div>Create the environment variable NFXTEXTFONT to specify the text font file. For example:</div> <div>NFXTEXTFONT=/opt/nms/nfx/text.fnt<br/>export NFXTEXTFONT</div> <div>Create the environment variable NFXHEADERFONT to specify the header font file. For example:</div> <div>NFXHEADERFONT=/opt/nms/nfx/header.fnt<br/>export NFXHEADERFONT</div> |                   |                                     |                |                  |   |   |                |   |
| C                 | <i>/.login</i>   | <div>Create the environment variable NFXTEXTFONT to specify the text font file. For example:</div> <div>setenv NFXTEXTFONT /opt/nms/nfx/text.fnt</div> <div>Create the environment variable NFXHEADERFONT to specify the header font file. For example:</div> <div>setenv NFXHEADERFONT /opt/nms/nfx/header.fnt</div>                                   |                   |                                     |                |                  |   |   |                |   |
| 5                 | Save the modified shell script file and exit the text editor.  |   |                   |                                     |                |                  |   |   |                |   |
| 6                 | For changes to the user's environment parameters to take effect in every UNIX window the user is logged onto, log out and then log back in as the designated user.   |   |                   |                                     |                |                  |   |   |                |   |

Refer to Modifying the Natural Access configuration file for information on completing the installation.

## Modifying the Natural Access configuration file

This topics describes:

- The services typically used by a NaturalFax application
- A sample Natural Access configuration file

Define all services that the application uses by specifying the service and service manager names in the Natural Access configuration file (*cta.cfg*). For more information, refer to the *Natural Access Developer's Reference Manual*.

To develop and run a NaturalFax application without specifying its services in *cta.cfg*, but you must explicitly define the services and service managers in your application code. For more information, refer to Initializing NaturalFax.

## NaturalFax services

A NaturalFax application typically uses the following services:

| This service... | With this service manager... | Provides the...   |
|-----------------|------------------------------|---|
| NFX             | NFXMGR                       | NaturalFax API.   |
| FAX             | FAXMGR                       | Group 3 fax protocol (ITU T.30).  |
| FXM             | ADIMGR                       | Interface to the hardware for sending or receiving a fax.   |
| NCC             | ADIMGR                       | Natural Access call control API.  |
| OAM             | OAMMGR                       | Functions for configuring, monitoring, and testing boards. Natural Access Server ( <i>ctdaemon</i> ) must be running. |

The default *cta.cfg* file includes the ADI and OAM services. Ensure that *cta.cfg* also includes the NFX, FAX, and FXM services. *cta.cfg* is located in the following directories:

| Operating system | Directory                          |
|------------------|------------------------------------|
| Windows          | <code>\nms\ctaccess\cfg</code>     |
| UNIX             | <code>/opt/nms/ctaccess/cfg</code> |

An application performing only fax file image conversions, and not fax transmission or reception, does not require the FAX or FXM services.

## Sample Natural Access configuration file

The following sample Natural Access configuration file shows the references to the NaturalFax services in bold:

```

=====
# cta.cfg
#
# This is an example of a file that specifies Natural Access configuration.
# It allows you to:
#   - Specify generic operational settings that apply to Natural Access
#     Server, ctdaemon, and all Natural Access applications.
#     Note: these settings can be overwritten by a Natural Access
#     application via ctaInitialize.
#   - Specify application specific settings
#   - Specify Natural Access Server and ctdaemon specific settings
#   - Redefine service specific parameter defaults
#
=====

#=====
# Natural Access System Configuration (ctasys)
#
# Valid options are:

```

```

# Service = name, dll          - Tells ctdaemon about available "services"
#                               - Tells the Natural Access Server what
#                               "services" to export
#
# Note: NCC should always precede ADI when both services are listed.
#=====
[ctasys]
Service = ncc, adimgr
Service = adi, adimgr
Service = dtm, adimgr
Service = ppx, ppxmgr
Service = swi, swimgr
Service = vce, vcemgr
Service = oam, oammgr

Service = nfx, nfxmgr # add this line for NaturalFax
Service = fax, faxmgr # add this line for NaturalFax

Service = fxm, adimgr # add this line for NaturalFax with
                      # AG boards or CG boards
Service = fxm, qdimgr # add this line for NaturalFax with
                      # QX 2000 boards

#=====
# Natural Access application configuration (application)
#
#=====
[application]
# DefaultServer = {inproc | localhost | server_name}
DefaultServer = inproc

# ContextNameUniqueness = {0|1} - Specification of responsibility for
#                               the context name uniqueness
#                               (application or Natural Access Server)
# 0 - Natural Access Server is responsible for the context name uniqueness.
#     Allow context name modification by Natural Access Server to ensure
#     name uniqueness (default)
# 1 - Application is responsible for the context name uniqueness.
#     Disallow context name modification by Natural Access Server,
#     uniqueness error expected from Natural Access Server
ContextNameUniqueness = 0

SessionTimeout = 3

#=====
# Natural Access Server configuration (server)
#
#=====
[server]
# ParmFlags = {0|1|2} - Natural Access Server parameter flags
# 0 - Parameter management done by Natural Access Server, ignore parameters
#     in ctdaemon (default).
# 1 - Parameter management done by ctdaemon
#     (requires ctdaemon is running, otherwise Natural Access Server exits).
# 2 - Parameter management done by ctdaemon if running, otherwise done by
#     Natural Access Server.
ParmFlags = 0

# TraceMode = {0|1|2} - Natural Access Server trace mode
# 0 - Tracing is disabled is default mode
# 1 - Enable tracing, immediate write to trace memory (default mode)
#     (requires ctdaemon to be running)
# 2 - Write trace memory on error only, keep local trace memory for each
#     context
TraceMode = 1

# Remote tracing parameters
# TraceMask = mask - Defines the default tracing mask
# TraceFile = filename - Defines trace file name
# StartCtaServer = {0|1} - 0 disables starting of built-in
#                           Natural Access Server

```



```

TraceMask = 0
# TraceFile = cta.log
StartCtaServer = 1

#
# DaemonAutoRestart parameter
# 0 - Do not automatically restart daemon after crash
# 1 - Automatically restart daemon after crash
#
DaemonAutoRestart = 0

# OAM Parameter
# 0 Configuration is not for an autonomous board
# 1 Assumes configuration is for autonomous board
AutonomousBoard = 0

=====
# Natural Access Default Parameter Changes (ctapar)
#
# Valid syntax for changes is:
#   service[.x].category.field = value
#
=====
[ctapar]

# by default, no changes are made

=====
[eof]

```

## Modifying the board keyword file

NMS OAM uses board keyword files to load and initialize all AG and CG boards in your system. To use NaturalFax, modify the board keyword files to:

- Explicitly allocate the DSPs on the board to perform fax functions.
- Specify the NaturalFax runtime software to be downloaded at board initialization.

Use the sample board keyword files provided with NaturalFax as models. Choose a board keyword file that corresponds to the hardware installed in your system.

| For more information about...         | Refer to...   |
|---------------------------------------|---|
| Fax performance                       | Strategies for optimizing performance.                  |
| NMS OAM                               | The <i>NMS OAM System User's Manual</i> .               |
| The keywords specified for each board | The board-specific installation and developer's manual. |

The NMS OAM configuration files invoke the board keyword files. The NMS OAM configuration files are located in the following directories:

| Operating system | Directory   |
|------------------|---|
| Windows          | <i>\nms\ag\cfg</i><br><i>\nms\oam\cfg</i>         |
| UNIX             | <i>/opt/nms/ag/cfg</i><br><i>/opt/nms/oam/cfg</i> |

The following table lists some of the sample board keyword files for boards that support NaturalFax.

**Note:** These board keyword files are valid for CompactPCI, PCI, and PCIe (PCI Express) boards. When using the NMS OAM configuration file on a CompactPCI board, change the product name in the file to the board name (for example, AG\_2000C).

| Board       | Board keyword file        | Description  |
|-------------|---------------------------|--|
| AG 2000/200 | <i>agpi2fax.cfg</i>       | mu-law fax transmit and receive configuration file |
| AG 2000/400 | <i>agpi2faxes.cfg</i>     | A-law fax transmit and receive configuration file  |
| AG 2000-BRI | <i>agpi2bri_fax.cfg</i>   | mu-law fax transmit and receive configuration file |
|             | <i>agpi2bri_faxes.cfg</i> | A-law fax transmit and receive configuration file  |
| CG 6060     | <i>cg6060faxes.cfg</i>    | A-law fax and IVR configuration file               |
| CG 6565     | <i>cg6565fax.cfg</i>      | mu-law fax and IVR configuration file              |

## Configuring AG boards

Complete the following steps to modify the board keyword file to run NaturalFax on AG boards:

| Step | Action  |
|------|---|
| 1    | Verify that the DLMFiles keyword includes <i>ag2fax.leo</i> :<br><pre>DLMFiles[x] = ag2fax.leo</pre>  |
| 2    | Verify that the DSP.C5x[x].Image keyword includes the following files:<br>For mu-law:<br><pre>DSP.C5x[x].Image = ag2fax.c54</pre><br>For A-law:<br><pre>DSP.C5x[x].Image = ag2faxes.c54</pre> |
| 3    | Save the changes to the board keyword file and exit the text editor.  |
| 4    | Run NMS OAM to configure and initialize the AG board(s) as specified in the <i>oamsys.cfg</i> file.   |
| 5    | Refer to the sample board keyword files for AG 2000 boards.   |

## Configuring CG boards

Complete the following steps to modify the board keyword file to run NaturalFax on CG boards:

| Step | Action  |
|------|---|
| 1    | Verify that the DLMFiles[x] keyword includes the board keyword file.<br>For CG 6565, CG 6565C, and CG 6565E boards:<br><pre>DLMFiles[x] = cg6565fax</pre><br>For CG 6060/C boards:<br><pre>DLMFiles[x] = cg6060fax</pre>  |
| 2    | Verify that the DSP.C5x[x].Files keyword identifies <i>nmsfax</i> as one of the data processing modules (DPMs) loaded on specific DSPs.<br>The following example is for a CG 6x6x board:<br><pre>DSP.C5x[1..31].Files = nmsfax voice tone dtmf echo callp ptf mf</pre>  |
| 3    | Verify that the Resource[0].Definitions keyword identifies <i>nmsfax</i> as one of the data processing functions (DPFs) loaded on the DSPs.<br>The following example is for a CG 6x6x board:<br><pre>Resource[0].Definitions = (nmsfax   (dtmf.det_all &amp; ptf.det_2f &amp; voice.rec_24 &amp;                                 (voice.play_24_100   tone.gen)))</pre> |

| Step | Action   |
|------|--|
| 4    | Save the changes to the board keyword file and exit the text editor.                                       |
| 5    | Run NMS OAM to configure and initialize the CG board or boards as specified in the <i>oamsys.cfg</i> file. |

For more information, refer to the Sample board keyword files.

## Sample board keyword files

This topic presents sample board keyword files for running NaturalFax

### Sample AG 2000/200 board keyword file

The following sample board keyword file sets up an AG 2000/200 board for NaturalFax using mu-law:

```
#-----
#                               agpi2fax.cfg
#
# --- Alliance Generation Plug-Ins Configuration ---
#
# Sample configuration file for AG-2000/200 or AG-2000/400 with
# NaturalFax
#
# Board settings:
#   Product = AG_2000
#-----

# Uncomment the appropriate Trunk Control Protocol(s)

TCPFiles[0] = nocc.tcp      # "no trunk control" protocol
TCPFiles[1] = lps0.tcp      # Loopstart protocol

DLMFiles = gtp.leo voice.leo svc.leo ag2fax.leo
DSP.C5x[0].Image = ag2fax.c54

XLaw = MU-LAW

Clocking.HBus.ClockSource = OSC
Clocking.HBus.ClockMode   = STANDALONE

DSP.C5x[1..3].Files = mf signal callp tone dtmf ptf voice
# Default DSP file for AG 2000
```

### Sample CG 6565 board keyword file

The following sample board keyword file sets up an CG 6565 board for NaturalFax using mu-law:

```
#=====
#   CG6565fax.cfg
#
#   This file configures the board to run 120 ports of Voice\xfax only
#
#   Note: This configuration file configure board just to do Fax with CAS.
#         If you like to run NOCC with ISDN/CAS, please modify this file.
#=====
#
#   cg6565fax.cfg
#   CG 6565 configuration file
#
#   This file configures the board to run FAX with T1 (E1 commented out)
#
```

```

Clocking.HBus.ClockMode           = STANDALONE
Clocking.HBus.ClockSource         = OSC
# use of DSP.C5x[x].Os is recommended
DSP.C5x[0..95].Os                 = dspos6u
# *****
# NOTE: E1 configuration is commented out. If you want E configuration, uncomment below,
#       and comment out OR remove T1 configuration
# NOTE: Adjust idle code and TCP Files for the country you are using accordingly
# *****

# *****
# E1 configuration (! commented out by default)
# *****
#DSPStream.VoiceIdleCode[0..15]    = 0xD5
#DSPStream.SignalIdleCode[0..15]   = 0x09

#NetworkInterface.T1E1[0..15].Type = E1
#NetworkInterface.T1E1[0..15].Impedance = G703_120_OHM
#NetworkInterface.T1E1[0..15].LineCode = HDB3
#NetworkInterface.T1E1[0..15].FrameType = CEPT
#NetworkInterface.T1E1[0..15].SignalingType = CAS

#DSP.C5x[0..95].Libs[0]            = cg6kliba
#DSP.C5x[0..95].XLaw               = A_LAW
# *****
# T1 configuration
# *****
DSPStream.VoiceIdleCode[0..15]     = 0x7F
DSPStream.SignalIdleCode[0..15]    = 0x00

NetworkInterface.T1E1[0..15].Type  = T1
NetworkInterface.T1E1[0..15].Impedance = DSX1
NetworkInterface.T1E1[0..15].LineCode = B8ZS
NetworkInterface.T1E1[0..15].FrameType = ESF
NetworkInterface.T1E1[0..15].SignalingType = CAS

DSP.C5x[0..95].Libs[0]             = cg6klibu
DSP.C5x[0..95].XLaw                = MU_LAW
# *****
#-----
# Hardware Echo Cancellation
# NOTE: it is in by pass by default
# NOTE: uncomment the following two keyword lines to enable and set the XLaw accordingly
#-----
# HardwareEcho.EchoChipEnabled = YES
# HardwareEcho.XLaw = A_LAW

#-----
# Resource management
#-----
Resource[0].Name                    = RSC0
Resource[0].Size                    = 120
# *****
# NOTE: E1 configuration is commented out. If you want E configuration, uncomment below,
#       and comment out OR remove T1 configuration
# *****
#Resource[0].TCPs                   = mfc0 nocc
Resource[0].TCPs                    = wnk0 nocc
Resource[0].StartTimeSlot           = 0

#-----
# Before modifying this resource definition string refer to the CG6565
# Installation and Developers Manual.
#-----

#-----
# FAX OR IVR (Definition can be used with nfxsend or nfxrecv type of applications)
#-----

Resource[0].Definitions              = (nmsfax | (dtmf.det_all & ptf.det_2f & \

```

```

voice.rec_24 & (voice.play_24_100 | tone.gen)))

#-----
# FAX and IVR (Definition can be used with caller or faxback type of applications)
# NOTE: Changing to following resource definition will/might decrease port count
#-----
#Resource[0].Definitions = ( dtmf.det_all & ptf.det_2f & ( nmsfax | \
#(voice.rec_24 & (voice.play_24_100 | voice.play_24_150 | voice.play_24_200)) | tone.gen ))

Resource[0].DSPs = \
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 \
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 \
64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95
DebugMask = 0x0
DLMFiles[0] = cg6565fax

```

## Sample CG 6060 board keyword file

The following sample board keyword file sets up an CG 6565 board for NaturalFax using A-law:

```

#=====
# CG6060fax.cfg
#
# This file configures the board to run Voice and Fax with CAS
#
# Board settings:
# Product = CG_6060 or CG_6060C
#
#=====

#-----
# Clock configuration
#-----
Clocking.HBus.ClockMode = STANDALONE
Clocking.HBus.ClockSource = OSC
#Clocking.HBus.ClockSourceNetwork = 1
#-----
# E1 configuration
#-----

NetworkInterface.T1E1[0..15].Type = E1
NetworkInterface.T1E1[0..15].Impedance = G703_120_OHM
NetworkInterface.T1E1[0..15].LineCode = HDB3
NetworkInterface.T1E1[0..15].FrameType = CEPT
NetworkInterface.T1E1[0..15].SignalingType = CAS

DSPStream.VoiceIdleCode[0..15] = 0xD5
DSPStream.SignalIdleCode[0..15] = 0x0D

DSP.C5x[0..95].Libs = cg6kliba
DSP.C5x[0..95].XLaw = A_LAW

#-----
# T1 configuration (commented out)
#-----
# NetworkInterface.T1E1[0..15].Type = T1
# NetworkInterface.T1E1[0..15].Impedance = DSX1
# NetworkInterface.T1E1[0..15].LineCode = B8ZS
# NetworkInterface.T1E1[0..15].FrameType = ESF
# NetworkInterface.T1E1[0..15].SignalingType = CAS

# DSPStream.VoiceIdleCode[0..15] = 0x7F
# DSPStream.SignalIdleCode[0..15] = 0x00
# DSP.C5x[0..95].Libs = cg6klibu
# DSP.C5x[0..95].XLaw = MU_LAW
#-----
# Resource Management
#-----
Resource[0].Name = RSC0
Resource[0].Size = 240

```

```

Resource[0].TCPS                                = nocc wnk0

#-----
# Before modifying this resource definition string refer to the CG6060 or
# CG6060C Installation and Developers Manual.
#-----

Resource[0].Definitions                         = (dtmf.det_all & ptf.det_2f & \
  (nmsfax | \
    (tone.gen | voice.rec_24 | \
      (voice.play_24_100 | voice.play_24_150 | voice.play_24_200))))

Resource[0].Dsps = 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 \
  23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 \
  48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 \
  72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95

DLMFiles[0]                                     = cg6060fax
DebugMask                                       = 0x0

```

## Verifying NaturalFax

After you install and configure the NaturalFax software, verify that the system is operational before you continue.

### Verifying the NaturalFax library

Use the Natural Access version checker utility, *ctavers*, to verify that the NaturalFax software can be successfully loaded. The Natural Access version checker verifies that all the Natural Access libraries defined in the Natural Access configuration file (*cta.cfg*) are accessible. The version checker program displays a list of the versions of all the Natural Access components.

Refer to the [Sample Natural Access configuration file](#) for more information.

To run *ctavers*, enter:

```
ctavers
```

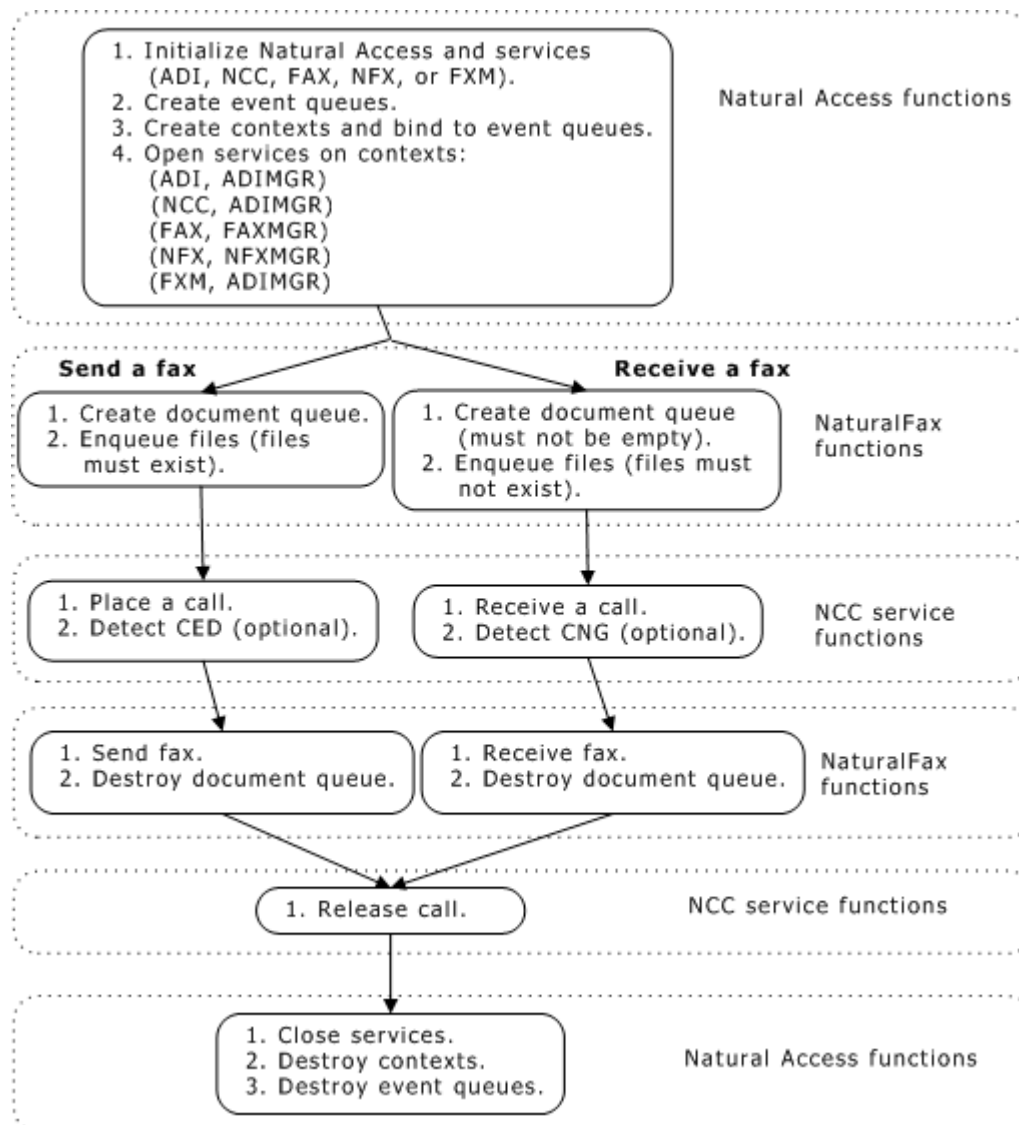
### Sending and receiving a fax

Use *nfxsend* and *nfxrecv* to send and receive a test fax. You can use the sample TIFF\_F file, *sample.tif*, to verify your installation.

## 4. Developing applications

### A typical fax application

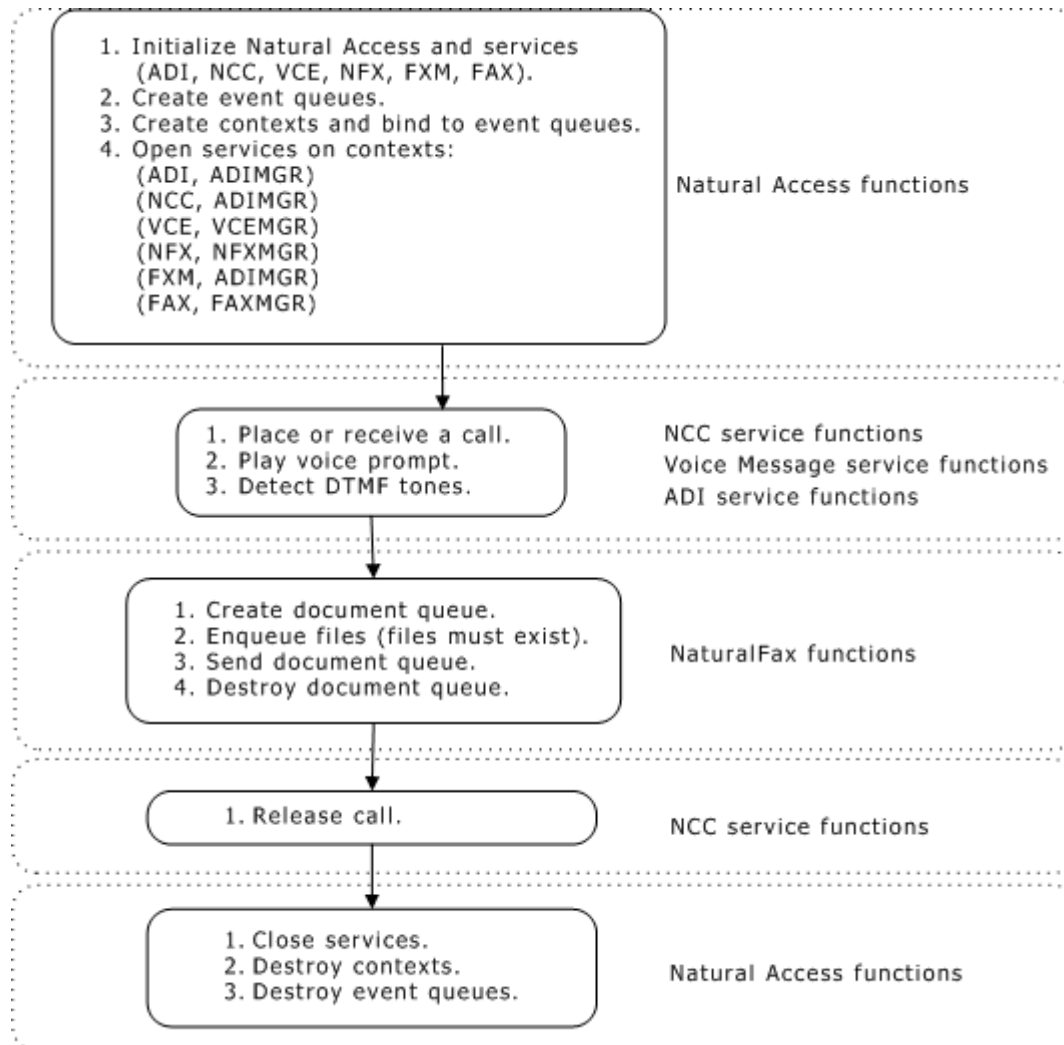
The following illustration shows the functions involved in a typical application that can transmit or receive faxes. NaturalFax invokes functions for transmitting and receiving a fax when a call is in the connected state. The document queue can be destroyed as soon as the fax session is complete, or after the call is disconnected.





## A typical fax and voice application

The following illustration shows a typical fax and voice application. In this application, the document queue is created when the call is in the connected state.



## Setting up the Natural Access environment

Before calling functions from the NaturalFax library, complete the following steps to set up the Natural Access environment:

| Step | Action                            |
|------|-----------------------------------|
| 1    | Initialize NaturalFax.            |
| 2    | Create event queues and contexts. |
| 3    | Open services on each context.    |

Each of these steps is described in the following sections. For general information on setting up the Natural Access environment, refer to the *Natural Access Developer's Reference Manual*.

## Initializing NaturalFax

Use the **ctaInitialize** Natural Access function to initialize NaturalFax and register the services available to the application. If *ctdaemon* is running, all services defined in *cta.cfg* are automatically registered when you invoke **ctaInitialize**.

If *ctdaemon* is not running, you must explicitly define the services and service managers in the call to **ctaInitialize**, as shown in the following code sample:

```
static DWORD initialize_base (void)
{
    CTA_SERVICE_NAME init_services [] =
    {
        { "NCC", "ADIMGR" },
        { "ADI", "ADIMGR" },
        { "FXM", "ADIMGR" },
        /* { "ADI", "QDIMGR" }, for QX 2000 boards */
        /* { "FXM", "QDIMGR" }, for QX 2000 boards */
        { "FAX", "FAXMGR" },
        { "NFX", "NFXMGR" },
    };
    CTA_INIT_PARMS init_parms =
    {
        sizeof (CTA_INIT_PARMS),
    };
    DWORD status;
    static int initialized = 0;
    if (initialized)
        return SUCCESS;
    status = ctaInitialize (init_services,
                           sizeof init_services / sizeof init_services [0],
                           & init_parms);
    if (status != SUCCESS)
        show_error (NULL_CTAHD, status, "Natural Access initialization
        failed");
    else
        initialized = 1;
    return status;
}
```

Refer to Modifying the Natural Access configuration file for a list of NaturalFax services.

## Creating event queues and contexts

After initializing NaturalFax, complete the following steps to create the event queues and contexts:

| Step                               | Action  |                 |         |                                    |     |                                    |     |        |     |        |     |
|------------------------------------|---|-----------------|---------|------------------------------------|-----|------------------------------------|-----|--------|-----|--------|-----|
| 1                                  | <p>Create one or more event queues by calling <b>ctaCreateQueue</b> and specifying the service managers to attach to each queue. Attaching or binding a service manager to a queue makes that service manager available to the queue. A NaturalFax application uses the following service managers:</p> <table> <tr> <th>Service manager</th><th>Service</th></tr> <tr> <td>ADIMGR (QDIMGR for QX 2000 boards)</td><td>NCC</td></tr> <tr> <td>ADIMGR (QDIMGR for QX 2000 boards)</td><td>FXM</td></tr> <tr> <td>NFXMGR</td><td>NFX</td></tr> <tr> <td>FAXMGR</td><td>FAX</td></tr> </table> | Service manager | Service | ADIMGR (QDIMGR for QX 2000 boards) | NCC | ADIMGR (QDIMGR for QX 2000 boards) | FXM | NFXMGR | NFX | FAXMGR | FAX |
| Service manager                    | Service   |                 |         |                                    |     |                                    |     |        |     |        |     |
| ADIMGR (QDIMGR for QX 2000 boards) | NCC   |                 |         |                                    |     |                                    |     |        |     |        |     |
| ADIMGR (QDIMGR for QX 2000 boards) | FXM   |                 |         |                                    |     |                                    |     |        |     |        |     |
| NFXMGR                             | NFX   |                 |         |                                    |     |                                    |     |        |     |        |     |
| FAXMGR                             | FAX   |                 |         |                                    |     |                                    |     |        |     |        |     |
| 2                                  | <p>Create a context by calling <b>ctaCreateContext</b>, passing the queue handle (<b>ctaqueuehd</b>) returned from <b>ctaCreateQueue</b>. All events for services on the context are received in the specified event queue.</p> <p><b>ctaCreateContext</b> returns a context handle (<b>ctahd</b>). The application supplies the context handle when invoking NaturalFax functions. Events communicated back to the application are also associated with the context.</p>   |                 |         |                                    |     |                                    |     |        |     |        |     |

Refer to the *Natural Access Developer's Reference Manual* for details on the programming models created by the use of Natural Access contexts and queues.

## Opening services

Opening a service on a context creates a service instance that defines a service and its associated resources (for example, MVIP board, stream, and timeslot). The context is a bound collection of service instances. A fax application usually opens the NFX, FAX, FXM, ADI, and NCC services on each individual context. Each context must specify the same board and MVIP address for the ADI and FXM service instances bound to it.

To open services on a context, call **ctaOpenServices** and pass a context handle and a list of service descriptor structures, one for each service. The service descriptor structure, **CTA\_SERVICE\_DESC**, contains the **CTA\_SERVICE\_NAME**, **CTA\_SERVICE\_ADDR**, **CTA\_SERVICE\_ARGS**, and **CTA\_MVIP\_ADDR** substructures, which provide the name of the service, its service manager, and service-specific arguments.

An application must open the NCC service to place or receive calls. The application must assign hardware resources to the service by filling in the board, bus, stream, timeslot, and mode fields in the **CTA\_MVIP\_ADDR** substructure. To ensure that the context is able to transmit and receive voice and signaling data, use **ADI\_FULL\_DUPLEX** mode.

The Fax Manager (FXM) service must use service-specific arguments to identify its hardware resources, because it provides the hardware interface for the NaturalFax service. When an application opens the FXM service, it must fill in the board, bus, stream, timeslot, and mode fields in the CTA\_MVIP\_ADDR substructure.

The NaturalFax (NFX) and Fax (FAX) services do not require any service-specific arguments. The NFX and FAX services use the same hardware resources specified by the FXM service opened on the same context.

The following code sample demonstrates creating a context and opening the ADI, NCC, FXM, FAX, and NFX services:

```
static DWORD create_context (CTAQUEUEHD queue,
                           DWORD      board,
                           DWORD      stream,
                           DWORD      timeslot,
                           DWORD      closure,
                           char       * contextname,
                           CTAHD      * contextptr)
{
    DWORD status;
    CTA_SERVICE_DESC services [4];
    DWORD service_count = 0;
    CTA_EVENT event;

    status = ctaCreateContext (queue, closure, contextname, contextptr);
    if (status != SUCCESS)
    {
        show_error (NULL_CTAHD, status, "Natural Access context creation
        failed");
        return status;
    }
    /*
    * Now that we have a valid context, we can use
    * ctaGetText to get proper error messages.
    */

    memset (& services, 0, sizeof services);

    services[service_count].name.svcname      = "ADI";
    services[service_count].name.svcmgrname    = "ADIMGR";
    /* for QX 2000 boards, use QDIMGR instead of ADIMGR */
    services[service_count].mvipaddr.board     = board;
    services[service_count].mvipaddr.stream    = stream;
    services[service_count].mvipaddr.timeslot  = timeslot;
    services[service_count].mvipaddr.mode      = ADI_FULL_DUPLEX;
    service_count += 1;

    services[service_count].name.svcname      = "NCC";
    services[service_count].name.svcmgrname    = "ADIMGR";
    /* for QX 2000 boards, use QDIMGR instead of ADIMGR */
    services[service_count].mvipaddr.board     = board;
    services[service_count].mvipaddr.stream    = stream;
    services[service_count].mvipaddr.timeslot  = timeslot;
    services[service_count].mvipaddr.mode      = ADI_FULL_DUPLEX;
    service_count += 1;

    services[service_count].name.svcname      = "FXM";
    services[service_count].name.svcmgrname    = "ADIMGR";
    /* for QX 2000 boards, use QDIMGR instead of ADIMGR */
    services[service_count].mvipaddr.board     = board;
    services[service_count].mvipaddr.stream    = stream;
    services[service_count].mvipaddr.timeslot  = timeslot;
    services[service_count].mvipaddr.mode      = ADI_FULL_DUPLEX;
    service_count += 1;

    /*
    * Specific MVIP_ADDR field assignments for the FXM, FAX, and NFX
    * services are ignored; these services use the MVIP_ADDR data that was
```

```

    * passed to the ADI service.
*/

services[service_count].name.svcname      = "FAX";
services[service_count].name.svcmgrname   = "FAXMGR";
services[service_count].mvipaddr.board    = board;
services[service_count].mvipaddr.stream   = stream;
services[service_count].mvipaddr.timeslot = timeslot;
services[service_count].mvipaddr.mode     = ADI_FULL_DUPLEX;
service_count += 1;

services[service_count].name.svcname      = "NFX";
services[service_count].name.svcmgrname   = "NFXMGR";
services[service_count].mvipaddr.board    = board;
services[service_count].mvipaddr.stream   = stream;
services[service_count].mvipaddr.timeslot = timeslot;
services[service_count].mvipaddr.mode     = ADI_FULL_DUPLEX;
service_count += 1;

status = ctaOpenServices (* contextptr, services, service_count);
if (status != SUCCESS)
{
    show_error (* contextptr, status, "open services failed");
    return status;
}

/* Now we need to wait for the CTAEVN_OPEN_SERVICES_DONE event. */
status = wait_event (queue, * contextptr, CTAEVN_OPEN_SERVICES_DONE,
                    NULL, 0, & event);
if (status != SUCCESS)
{
    show_error (* contextptr, status, "open services failed");
    return status;
}
else if (event.value != CTA_REASON_FINISHED)
{
    show_error (* contextptr, event.value, "open services done event");
    return event.value;
}
else
    return SUCCESS;
}

```

## Establishing a call

The process of establishing a call differs depending upon whether the application is acting as a calling fax terminal or as a called fax terminal. Use one or more call control functions from the NCC service to establish a call. Refer to the *Natural Call Control Service Developer's Reference Manual* for information on call control.

## Placing a call

Call progress analysis in the NCC service includes the capability to detect CED tones. A CED tone is a three second 2100 Hz tone indicating that a fax terminal has answered.

Enable CED tone detection when invoking the **nccPlaceCall** NCC service function. The application generates an NCC service call progress event if it detects a CED tone. If no CED tone is detected, the application provides a way to handle a call answered by a person or by a modem rather than by a fax machine.

## Receiving a call

The application invokes the **nccAnswerCall** NCC service function and the **adiStartToneDetector** ADI service function to answer an inbound call and detect CNG tones. The calling fax terminal sends a CNG tone. A CNG tone is a 0.5 second 1100 Hz tone that indicates a fax terminal is calling.

If an application needs to handle both fax and human callers, it should start a special tone detector to detect CNG tones in conversation state. The application starts a fax session only if a CNG tone is detected. In a fax-only application, a fax session may be started as soon as the call is answered.

## Using document queues

NaturalFax functions operate on document queues, not on individual files. Use a document queue for all fax operations, including transmitting or receiving single documents.

A document queue contains a list of one or more document files in TIFF-F or TIFF-S format. NaturalFax has two kinds of document queues: send queues and receive queues. Use send queues to transmit documents, and receive queues to receive documents.

Each document queue is linked to a specific context. Each context may have multiple send and receive queues.

## Building a document queue

To create a new document queue, call **nfxCreateQueue**. This function returns a queue handle that uniquely identifies the queue. When you call **nfxCreateQueue**, specify the type of queue to be created (send or receive).

After creating a send queue, use **nfxEnqueueDoc** to add document files to the queue. Only files that already exist can be added to a send queue. The same file can be placed in multiple send queues simultaneously.

NaturalFax sends each enqueued file as a separate message (refer to the Group 3 fax technology section for information about T.30 messages). For example, if you enqueue a three page file and a five page file, the remote fax machine receives a three page message followed by a five page message. NaturalFax will not send both files in a single eight page message.

NaturalFax does not modify or delete any document files when sending. Only received document files can be modified. Each document to be received must have an entry in the receive queue. When adding files using **nfxEnqueueDoc**, the receive name can not be NULL, and the file must not already exist.

NaturalFax receives each multiple page message into a single file. The next document in the queue is used only if the remote fax machine sends a new message. This method of file management is specific to computer-based fax applications.

## Transmitting and receiving faxes

This topic presents the following information:

- Transmitting faxes
- Receiving faxes
- Polling the called fax terminal
- Answering a poll request
- Resetting a document queue

Transmitting or receiving document queues constitutes an active fax session. A fax session can include the polling of a remote fax terminal or responding to a poll request from a remote fax terminal.

After a document queue is transmitted or received, the queue can be reset. Each page in a document in a send queue is flagged as sent when it is successfully transmitted.

**Note:** You must continue processing events during an active fax session. Make sure that file I/O intensive operations, such as TIFF-F or TIFF-S format verification, do not interfere with the handling of events and cause the fax session to time out. Events must be processed within three seconds.

### Transmitting faxes

Use **nfxSendFax** to transmit all the documents in a specified send queue. You can track which documents are successfully sent by monitoring events during transmission.

NaturalFax typically ends a fax session by transmitting a DCN (disconnect) frame to the remote fax terminal and sending an NFXEVN\_SESSION\_DONE event to the application. The application can release the call as soon as it receives the NFXEVN\_SESSION\_DONE event; it does not need to wait for a response from the remote fax terminal. The application must use NCC service functions to release the call after the DONE event is received.

You can also transmit a procedure interrupt request (PRI) frame to the receiving fax terminal after completing a send operation. PRI requests that the receiving fax terminal permit an operator action such as picking up the handset for voice. Refer to **nfxSendFax** for information about sending PRI requests.

### Receiving faxes

An application must create a receive queue before it can call **nfxReceiveFax** and receive a queue of documents. The receive queue can not be empty; it must contain one or more file names of nonexistent files. The files must not exist when the receive queue is created or when **nfxReceiveFax** is invoked.

NaturalFax stores each fax document received in a separate file, using the list of file names specified by the NaturalFax receive queue. If the sending fax terminal indicates that it is starting a new document and NaturalFax has already used all available empty file names in the receive queue, the received document is appended to the last file in the receive queue. This could result in a TIFF file with unused image attributes.

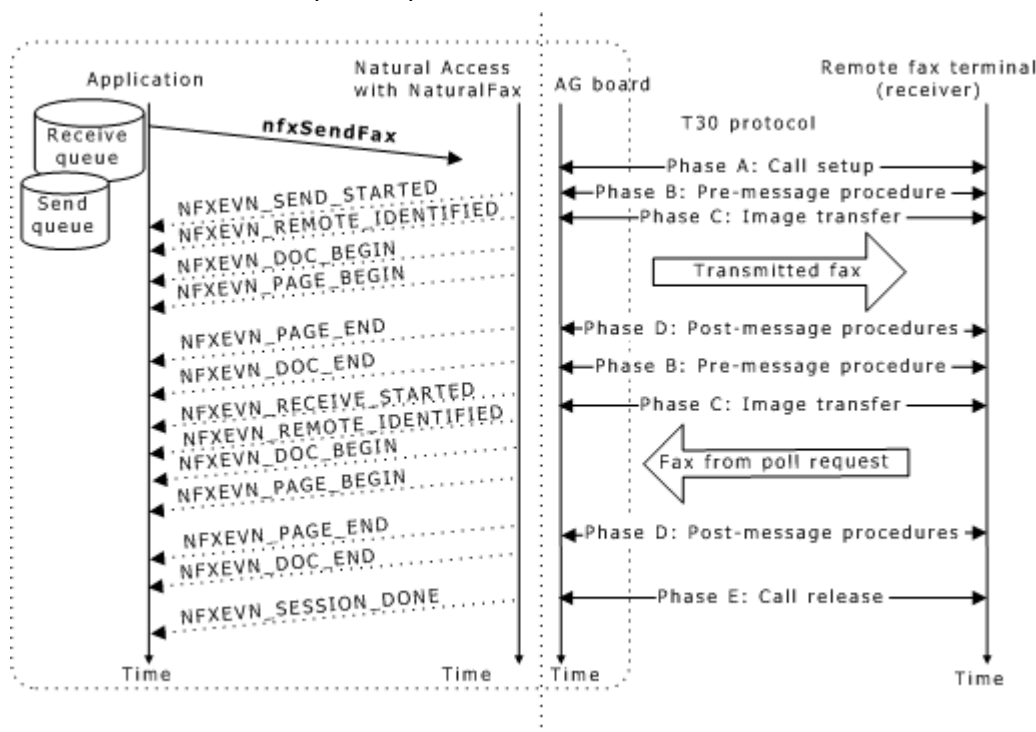
### Polling the called fax terminal

The fax polling operation enables the called fax terminal to send a queue of documents to the calling fax terminal. A caller can send any number of documents in a single queue and then poll the remote fax terminal. All documents in the send queue are transmitted before the called fax terminal can be polled.

To request polling, call **nfxSendFax** with a non-NULL **receive\_queue\_handle** argument. Since the calling fax terminal will be receiving faxes, it must have a receive queue ready to accept the received document files.

The **receive\_queue\_handle** argument controls polling. A call to **nfxSendFax** with a NULL **send\_queue\_handle** and a non-NULL **receive\_queue\_handle** initiates a fax session that polls the called fax terminal without transmitting any documents first.

The following illustration shows the exchanges between the application, the APIs, the hardware, and the remote fax terminal as an application transmits a fax, then receives the fax for which it sent a poll request:



## Answering a poll request

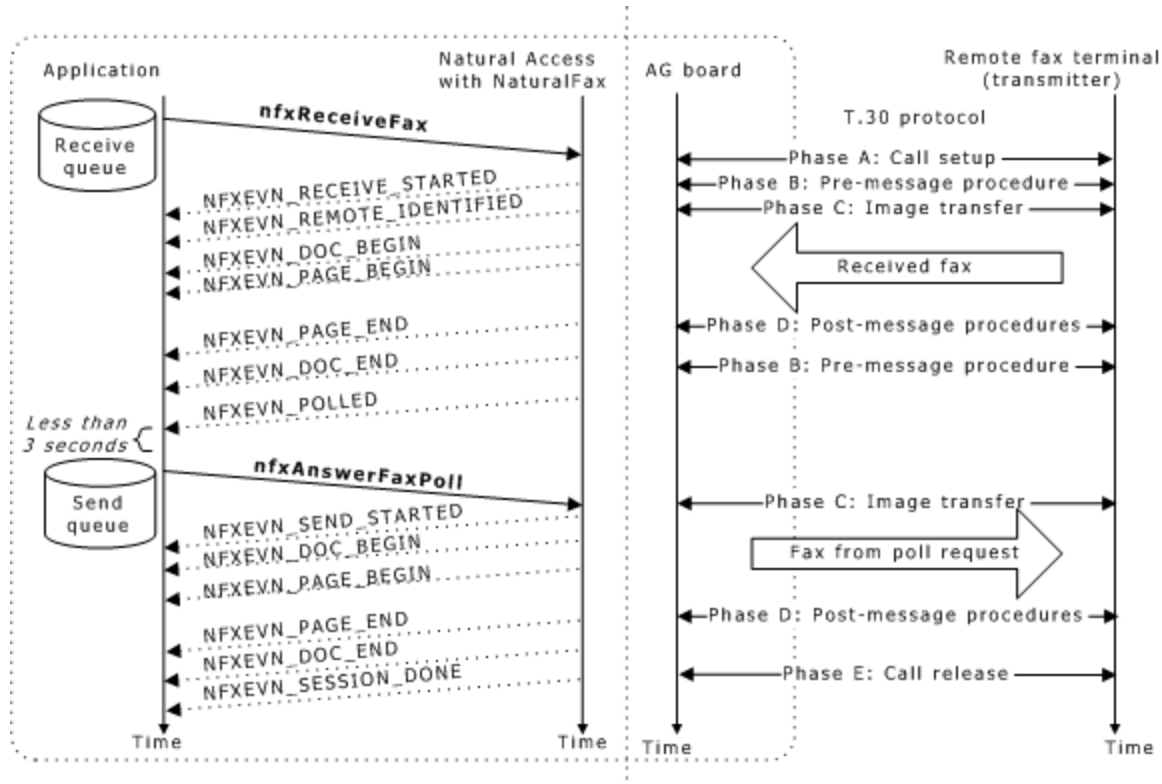
To enable polling from the receiving fax terminal, set `pollingenabled` to `NFX_YES` in `NFX_RECEIVE_PARMS`. The application is notified of a poll request when it receives an `NFXEVN_POLLED` event. Even with polling enabled, the application must decide if it will answer a poll request, and prepare a send queue or use an already-prepared send queue of documents.

An application uses **nfxAnswerFaxPoll** to transmit its send queue in response to a poll request, rather than **nfxSendFax**. **nfxAnswerFaxPoll** must be called within three seconds of the receipt of `NFXEVN_POLLED` event. If the application does not respond to the poll within three seconds, it receives `NFXEVN_SESSION_DONE`. Alternatively, you can call **nfxStopSession** to refuse the poll request from the remote fax terminal.

If polling is not enabled, the `NFXEVN_POLLED` event is not generated even if a poll request is made by the remote transmitter. The fax session then terminates.



The following illustration shows NaturalFax receiving a document, and then transmitting a document in response to a poll:



## Resetting a document queue

After a document queue is transmitted or received, use `nfxResetQueue` to reset it.

Each page in a document is flagged as sent when it is successfully transmitted. Resetting a send queue resets the flags for each page of each of its component documents to unsent.

Use one document queue per fax operation. For a fax broadcast application, use one send queue per broadcast session, and continue to reset the queue and resend its contents multiple times.

Resetting a receive queue changes the queue type to a send queue with all the pages flagged as unsent. This use is typical of a fax store-and-forward application. Once a receive queue is changed to a send queue, it cannot be changed back to a receive queue.

## Performing offline image conversion

You can convert the image characteristics of a file offline before transmitting a fax, or online while transmitting and receiving a fax. Because image conversion consumes significant processor time, you may want to perform file conversions offline to conserve CPU resources for processing events during an active fax session.

Use `nfxConvertFileDirect` to convert the image characteristics of a file offline. Specify the input file, the output file, and the desired image characteristics. Conversions of encoding type or page width do not affect image quality. Converting from a high resolution to a lower resolution will affect image quality. Converting an image from a lower to a higher resolution is not recommended, since the image cannot be improved and the resulting file is larger than the original.

NaturalFax supports TIFF-F files with different image formats on its pages by converting the format of all pages in the file to the format of the first page. If the file requires an image format other than that of the first page, use **nfxConvertFileDirect** to convert all the pages in a file to a specific image format. You can also use **nfxSplitFile** to split the original file into multiple files, each of which contains one or more pages that have the same image format. Splitting the file does not alter the image format of any of its pages. You may want to use **nfxSplitFile** to preserve the original image quality if a document has different resolutions on different pages.

Use **nfxMergeFile** to combine multiple pages stored in separate files into a single document for transmission. Merging the file does not alter the image format of any of its pages.

To verify that a document is in the correct format for transmission by NaturalFax, use **nfxCheckTIFF**.

File I/O intensive operations, such as TIFF-F or TIFF-S conversion or verification must not interfere with Natural Access event processing. Delays in application event processing can cause the fax session to time out. Applications must process events within three seconds of receipt. An application must continue processing events during an active fax session.

For more information, refer to Image format characteristics.

## Performing online image conversion

You can convert the image characteristics of a file offline before transmitting the fax, or online while transmitting and receiving a fax. Because image conversion consumes significant processor time, you may want to perform file conversions offline to conserve CPU resources for processing events during an active fax session.

NaturalFax's receive and transmit parameter structures, NFX\_RECEIVE\_PARMS and NFX\_TRANSMIT\_PARMS, contain image format parameters for encoding, resolution, and page width. These parameters are used with the OTFmode (on-the-fly) parameter to determine the format of the image being transmitted or received. Refer to Image format characteristics for more information.

Refer to NFX\_TRANSMIT\_PARMS or NFX\_RECEIVE\_PARMS for valid OTFmode parameters. When NaturalFax is receiving a fax, NFX\_DOC\_PARMS contains the image format parameters that determine how the image is stored.

NFX\_RECEIVE\_PARMS contains the badlineaction parameter, which controls how the receiving fax terminal manages bad lines of data. Refer to NFX\_RECEIVE\_PARMS for valid badlineaction parameters.

This topic presents the following information:

- Image conversion during fax transmission
- Image conversion during fax reception
- Generating TIFF-S files on receive

## Image conversion during fax transmission

NaturalFax uses the parameters from NFX\_TRANSMIT\_PARMS and the capabilities of the remote fax terminal to determine the image format used during a fax transmission.

The first step in determining the final transmission format is based on the capabilities of the transmitting and receiving fax terminals. The remote receiver announces its capabilities in the DIS frame. NaturalFax uses the image format parameters in the NFX\_TRANSMIT\_PARMS structure and the capabilities of the receiver to choose the initial transmission format. The initial transmission format is set to the NFX\_TRANSMIT\_PARMS value if the remote receiver can support it; otherwise it is set to the nearest value that the receiver supports.

The next step in determining the transmission format is based on the value of OTFmode in NFX\_TRANSMIT\_PARMS. The format of the stored file is compared with the initial transmission format, and conversion takes place according to the following criteria:

| If OTFmode is set to... | And the stored file format...   | Then...  |
|-------------------------|---|--|
| NFX_OTF_NEVER           | Requires capabilities less than or equal to the initial transmission format                 | The stored image format is selected as the final transmission format. No image conversion is performed.  |
| NFX_OTF_NEVER           | Requires more advanced capabilities than can be provided by the initial transmission format | The fax session fails, and NaturalFax returns NFXEVN_SESSION_DONE with the error NFXERR_INCOMPATIBLE_RECEIVER. No file is sent, since the system cannot satisfy the needs of the receiver without performing a conversion. No image conversion is performed. |
| NFX_OTF_ONLY_IF_FAIL    | Requires capabilities less than or equal to the initial transmission format                 | The stored file format is selected as the final transmission format. No image conversion is performed.   |
| NFX_OTF_ONLY_IF_FAIL    | Requires more advanced capabilities than can be provided by the initial transmission format | The file is converted to match the initial transmission format. The initial transmission format is selected as the final transmission format. NaturalFax performs on-the-fly image conversion.   |

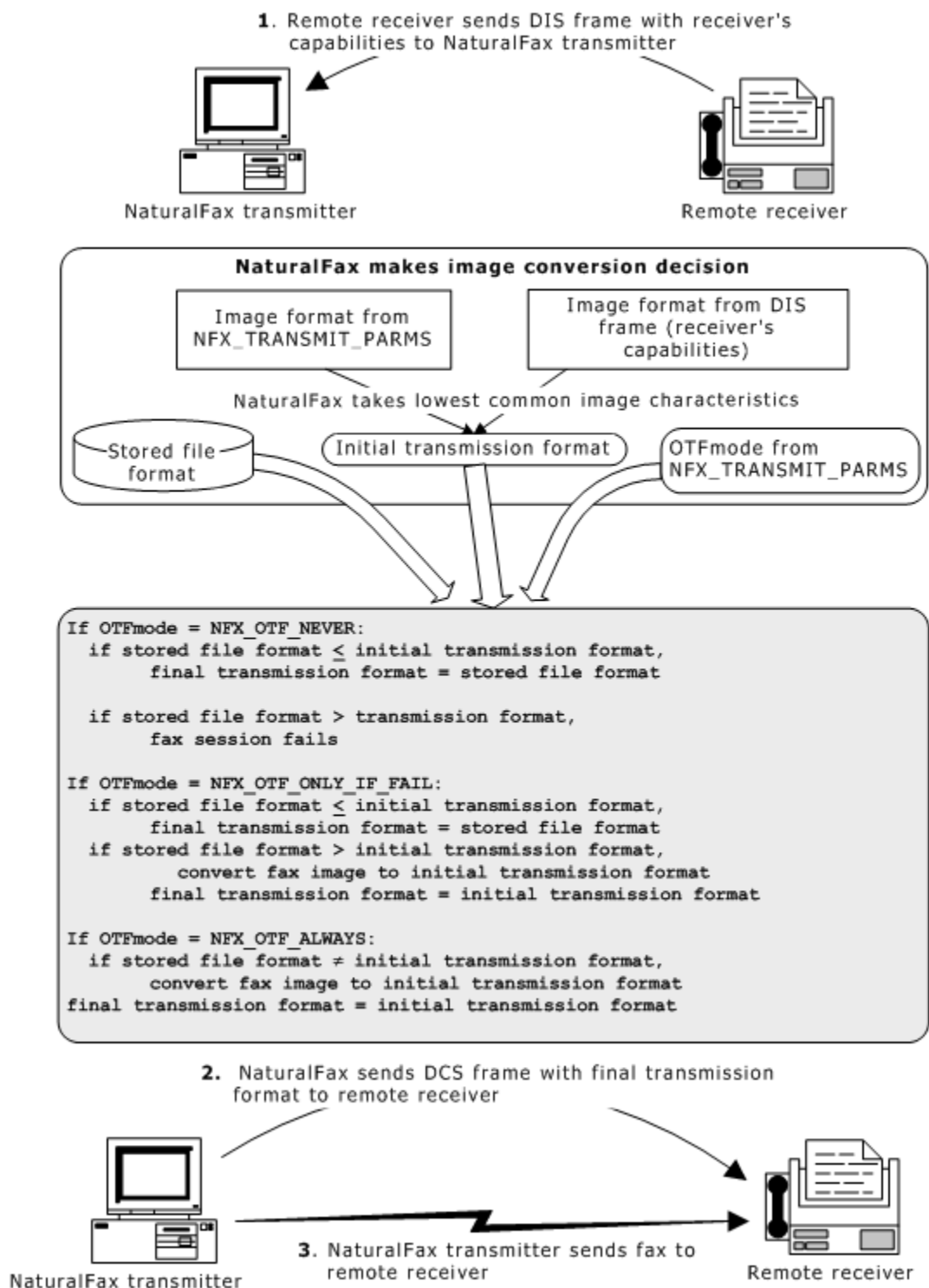
| If OTFmode is set to... | And the stored file format...   | Then...  |
|-------------------------|---|--|
| NFX_OTF_ALWAYS          | Does not match the initial transmission format, it will be converted. This mode will always use the encoding format characteristics specified by the initial transmission format. | There will only be a conversion for resolution and page width if it is necessary to support a lesser transmission capability. NaturalFax performs on-the-fly image conversion if required. |

After the decision is made, the final transmission format is sent back to the receiver in the DCS frame. The application can retrieve the final transmission format by interrogating the NFX\_DOC\_STATUS and NFX\_FAX\_STATUS data structures.

For the best image quality, an application should use the best formatting that the receiving fax machine can support. Set NFX\_TRANSMIT\_PARMs to use NFX\_ENCODE\_MMR, NFX\_RESOLUTION\_SUPER\_HIGH, and NFX\_PAGE\_WIDTH\_A3 to use the best capabilities of the receiving fax machine.

OTFmode can be set to NFX\_OTF\_ONLY\_IF\_FAIL to minimize the host execution time, or to NFX\_OTF\_ALWAYS to minimize transmission time. Checking and replacement of bad lines is performed when OTFmode is set to NFX\_OTF\_ONLY\_IF\_FAIL or NFX\_OTF\_ALWAYS.

The following illustration shows the image format decision process during fax transmission:

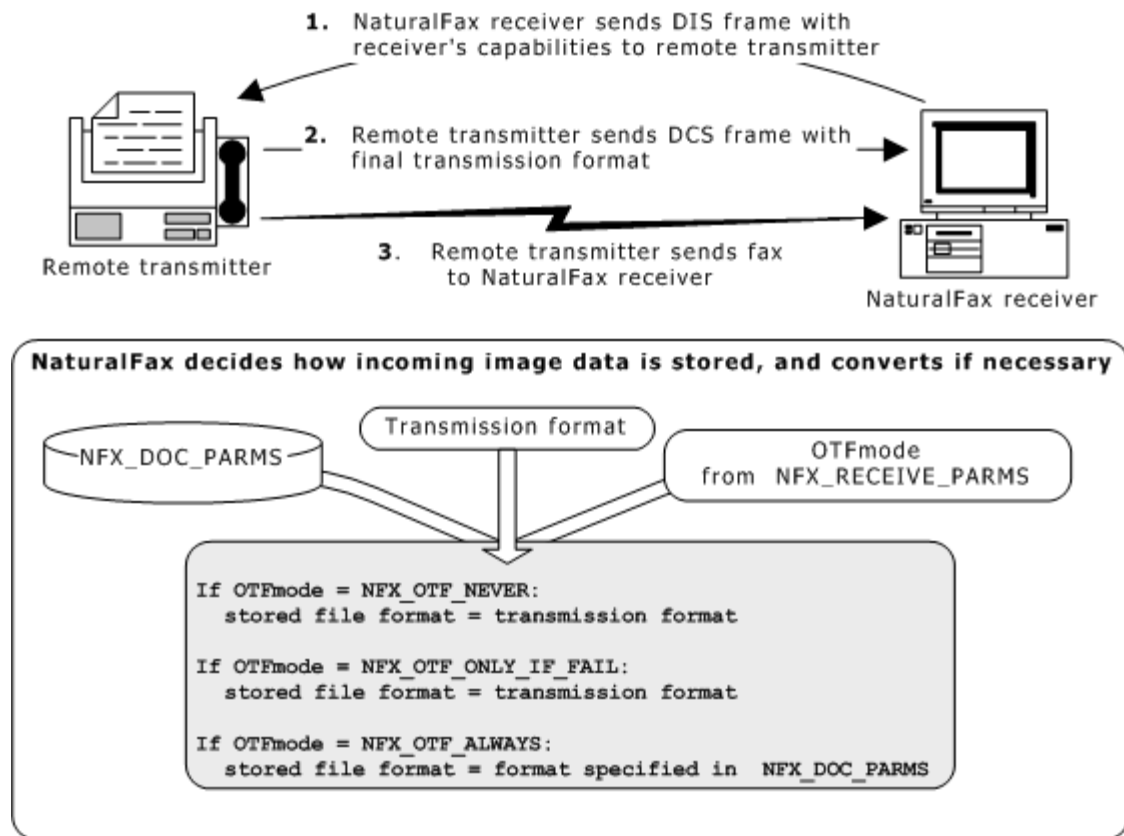


## Image conversion during fax reception

NaturalFax uses the parameters from NFX\_RECEIVE\_PARMS and the remote fax terminal's capabilities to determine the image format stored during a fax receive operation. This process includes the following steps:

| Step | Action   |
|------|--|
| 1    | When NaturalFax acts as the receiving fax terminal, it sends the image format values from NFX_RECEIVE_PARMS to the remote transmitter in a DIS frame to announce the receiver's capabilities to the transmitter.   |
| 2    | The transmitter sends back a DCS frame with the final transmission format. The remote transmitter controls the choice of the final transmission format.  |
| 3    | <p>Once the final transmission format is determined, the application uses the values from NFX_DOC_PARMS and the value for OTFmode in NFX_RECEIVE_PARMS to determine how the incoming image data is actually stored. Conversion only takes place if OTFmode is set to NFX_OTF_ALWAYS. Otherwise, the file is stored in the format received.</p> <p>If OTFmode is set to NFX_OTF_ALWAYS, the NFX_DOC_PARMS values for the document determine the final storage format. A conversion from lower to higher resolution, and from narrower to wider pages is performed if specified. Using NFX_OTF_ALWAYS is appropriate when your application requires a specific image format, and there are sufficient host CPU resources to support on-the-fly conversion.</p> <p>To use the minimum host CPU resources when receiving a fax, accept the highest quality the sender can provide. Save the image directly to a file without performing conversion using NFX_ENCODE_MMR, NFX_RESOLUTION_SUPER_HIGH, NFX_PAGE_WIDTH_A3, and NFX_OTF_NEVER in NFX_RECEIVE_PARMS.</p> |
| 4    | To retrieve the final storage format after the document is received, examine the NFX_DOC_STATUS or NFX_FAX_STATUS structures.  |

The following illustration shows how the stored image format is selected during a fax receive operation:



## Generating TIFF-S files on receive

When receiving an image to be used in future T.37 operations, the application can choose to:

- Ensure that the image is transmitted in TIFF-S format by advertising only TIFF-S compatible capabilities, which minimizes CPU resources
- Receive the image in the fastest means possible, and use the OTFmode to convert the image to TIFF-S format after reception.

The following table summarizes how the document will be transmitted, as determined by the OTFmode, when `NFX_RECEIVE_PARAMS.ENCODING = TIFF_S`.

| OTFmode        | Advertised parameters  |
|----------------|--|
| NFX_OTF_NEVER  | 1D encoding<br>Low resolution<br>A4 page width   |
| NFX_OTF_ALWAYS | MMR encoding (if ECM is enabled) or MR (if ECM is disabled)<br>Low resolution<br>A4 page width |

## Monitoring fax session status

This topic provides information on

- Tracing NaturalFax applications.
- Error handling during a fax session.

An application can actively monitor the progress of a fax session by explicitly querying the status, or passively by monitoring the NaturalFax events on a specified context.

When an application receives an error as part of an event, it can continue sending or receiving the remaining documents in the queue, or abort the fax session with **nfxStopSession**.

Use **nfxGetSessionStatus** to obtain a snapshot of the status of the current fax session. This function fills in an NFX\_FAX\_STATUS structure.

To retrieve information about a specific document, call **nfxGetDocStatus**. This function fills in an NFX\_DOC\_STATUS structure.

For more information, refer to:

- Alphabetical error summary
- Overview of events
- Overview of modem metrics

## Tracing NaturalFax applications

You can activate tracing for NaturalFax using Natural Access tracing functions. To use Natural Access tracing:

- *ctdaemon* must be running.
- *ctdaemon* must launch a trace thread. Refer to the *Natural Access Developer's Reference Manual* for more information on *ctdaemon*.
- The application must enable tracing in the call to **ctalInitialize**.

If the application enables tracing in **ctalInitialize** and *ctdaemon* is not running, **ctalInitialize** fails. The application must unset the trace mask in CTA\_INIT\_PARMS to disable tracing, and invoke **ctalInitialize** again.

- The application must initiate tracing by calling **ctaSetTraceLevel** with a specific context, the NFX service name, and a mask indicating what to trace.
- The NFX service must be specified in *cta.cfg*.

The NFX service logs internal trace information to *ctdaemon* according to the trace mask setting. The following NaturalFax trace masks are available:

| Trace mask       | Value    | Description                                 |
|------------------|----------|---|
| NFX_TRACE_T30    | 0x000100 | Logs information relating to T.30 protocol. |
| NFX_TRACE_CMDEVT | 0x000C00 | Logs all fax commands and events.           |
| NFX_TRACE_ALL    | 0x000F00 | Logs all available fax information.         |

Refer to the *Natural Access Developer's Reference Manual* for information about Natural Access trace masks.



The following code sample demonstrates setting the trace level:

```
ctaSetTraceLevel( ctahd, "NFX", NFX_TRACE_T30 | CTA_TRACEMASK_ALL_EVENTS );
```

## Error handling during a fax session

The application monitors all events during an active fax session (while sending or receiving a fax). Events provide information about error conditions, enabling the application to respond appropriately.

For information on NaturalFax errors returned as part of a NaturalFax DONE event, refer to the NaturalFax events. For information on Natural Access errors returned as part of a NaturalFax DONE event, refer to the *Natural Access Developer's Reference Manual*.

ADIEVN\_BOARD\_ERROR can be returned during an active fax session when an unexpected error occurs on your board. Contact NMS Communications with the specific error information. This should be considered a fatal error on this channel.

## Terminating and shutting down

After a fax session completes, the application must release the call and tear down or reset any document queues.

A normal fax session ends with the transmitter sending the DCN (disconnect) frame, signaling the remote fax terminal to disconnect. Some fax terminals disconnect the call before they receive or transmit the DCN frame. Under these conditions, the NaturalFax application receives an NCCEVN\_CALL\_DISCONNECTED event before it receives the NFXEVN\_SESSION\_DONE event.

The application should wait for the NFXEVN\_SESSION\_DONE event, and then release the call as usual. The NFXEVN\_SESSION\_DONE event contains a CTA\_REASON\_RELEASED or CTA\_REASON\_FINISHED reason code. Refer to [nfxReceiveFax](#) for more information.

A NaturalFax fax session terminates differently if the application terminating the fax is the:

- Natural Call Control service or NMS ISDN for Natural Call Control service (layer 4)
- NMS ISDN Messaging API

| Terminating application   | Termination type      | The application...   |
|---|-----------------------|--|
| Natural Call Control service or NMS ISDN for Natural Call Control | Normal                | Receives NFXEVN_SESSION_DONE with reason code CTA_REASON_FINISHED and issues <b>nccDisconnectCall</b> .  |
|   | Abnormal              | Receives NFXEVN_SESSION_DONE with reason code CTA_REASON_RELEASED (or other error) and issues <b>nccDisconnectCall</b> .   |
|   | Application-initiated | Calls <b>nfxStopSession</b> and waits for NFXEVN_SESSION_DONE with reason code CTA_REASON_STOPPED. It then calls <b>nccDisconnectCall</b> .  |
| NMS ISDN Messaging  | Normal                | Receives NFXEVN_SESSION_DONE with reason code CTA_REASON_FINISHED and issues a clear request (ACU_CLEAR_RQ).   |
|   | Abnormal              | Receives a clear indication (ACU_CLEAR_IN) and issues <b>nfxStopSession</b> . It then waits for NFXEVN_SESSION_DONE. The application issues a clear response (ACU_CLEAR_RS).<br><br>Or, the application receives a clear confirmation (ACU_CLEAR_CO) and issues <b>nfxStopSession</b> . It then waits for NFXEVN_SESSION_DONE. |
|   | Application-initiated | Calls <b>nfxStopSession</b> and waits for NFXEVN_SESSION_DONE with reason code CTA_REASON_STOPPED. The application issues a clear request (ACU_CLEAR_RQ).  |

For information about releasing and terminating calls, refer to the *ADI Service Developer's Reference Manual*, *NMS ISDN for Natural Call Control Developer's Manual*, and *NMS ISDN Messaging API Developer's Reference Manual*.

## Tearing down or resetting a document queue

A NaturalFax application can initiate multiple fax sessions during its execution, either simultaneously or in a series, according to the programming model used. It can also reset a document queue and use it again in a subsequent fax session.

Call **nfxDestroyQueue** to destroy a document queue.

To use a document queue after it successfully transmits or receives a fax, call **nfxResetQueue**. Resetting a send queue resets the flags for each page of each of its component documents from sent to unsent. Resetting a receive queue changes it to a send queue, with all its component documents flagged as unsent. Once a receive queue is changed to a send queue, it cannot be changed back to a receive queue.

## Closing Natural Access services

Natural Access supports opening and closing services on a context as needed during an application's execution. An application can free system resources it no longer needs to optimize performance.

Use **ctaCloseServices** to close the NaturalFax services. **ctaCloseServices** destroys any open document queues associated with the service on the specified context.

**ctaCloseServices** does not release the call, nor does it close the ADI service unless specified.

## 5. Optimizing performance

---

### Strategies for optimizing performance

A board's DSP resources can support a combination of IVR, fax transmit, and fax receive operations. IVR operations include basic Natural Access telephony functionality such as voice play and record, tone detection, and tone generation. You can control DSP resource allocation for a particular board by editing its configuration in the board keyword file assigned to the board.

Hardware-related performance is measured by the

- Available ports per board.
- Number and type of operations that can be run per port.

Refer to `NaturalFax_and_NMS_hardware` for a list of boards that support NaturalFax.

Use one of the following strategies to make the most efficient use of board resources, according to the design and purpose of the application:

- Maximize the number of ports that can run IVR and fax transmit operations.
- Maximize the number of universal ports on a board. A universal port can use any combination of IVR or fax functions. If a system is configured for universal ports, the application can use any combination of IVR functions such as voice play, voice record, tone detection, and tone generation with fax transmit or fax receive functions. The ability to alternate between IVR and fax functions within a single phone call is supported in a universal port configuration.
- Support all ports on a board

**Note:** For the purposes of this manual, universal port does not include speech vocoding functions.

The configurations presented in this section were load tested without echo cancellation, using speech encoding for voice functions. If your application requires a different use of DSP resources (for example, adding echo cancellation or using different speech encoding), you may achieve different performance results.

Refer to the appropriate sample files, as listed in each table, for detailed comments and specific configuration statements to achieve the desired performance.

In some configurations, a board's DSP resources cannot support an application that uses all of the ports on the board so some ports are left idle. Each DSP core can support a maximum of eleven fax-only operations. If the number of operations performed exceeds the board's ability to support them, an `NFXEVN_SESSION_DONE` event is returned with the reason `CTAERR_OUT_OF_RESOURCES`. This may cause problems to fax sessions that are in progress.

NMS recommends that the number of fax operations not go above the number of fax operations allowed (number of DSP cores loaded with fax multiplied by eleven).

Test your application carefully using the number of ports and the functionality that it will use during normal operation to ensure you have sufficient DSP resources and host CPU resources.

Refer to the board-specific installation and developer's manual for further information on DSP requirements. Refer to the DSP requirements for the following boards:

- AG boards
- CG boards

## Maximizing ports for fax transmission

The following table shows the maximum number of IVR and fax transmit ports that specific boards can support. The table also lists the example configuration files.

This performance strategy achieves the highest number of ports per board, but all ports may not support all operations. Some of the hardware configurations listed can support only IVR and fax transmit operations, and cannot support fax receive operations on any ports. If you need fax receive capability, configure the system for maximizing universal ports.

| Maximum fax transmit ports per board |       |              | Configuration file for... |                          |
|--------------------------------------|-------|--------------|---------------------------|--------------------------|
| Board                                | Ports | Restrictions | mu-law                    | A-law                    |
| AG 2000/200                          | 8     | None         | <i>agpi2fax.cfg</i>       | <i>agpi2faxa.cfg</i>     |
| AG 2000-BRI                          | 8     | None         | <i>agpi2bri_fax.cfg</i>   | <i>agpi2bri_faxa.cfg</i> |
| AG 2000/400                          | 16    | None         | <i>agpi2fax.cfg</i>       | <i>agpi2faxa.cfg</i>     |

The AG 2000/400 board provides DSPs for up to 16 ports of IVR and fax, but only eight line interfaces. Eight additional line interfaces must be provided with additional hardware.

**Note:** These calculations are valid for both CompactPCI and non-CompactPCI boards.

## Maximizing universal ports

The following table shows the maximum number of universal ports that specific boards can support. The table also lists the example configuration files.

Universal ports can run any IVR or fax operation. In some configurations, using the maximum number of universal ports means none of the remaining ports can be used. Under these conditions, each available port can support all operations, but all ports on the board may not be available. This performance strategy provides the best functionality per port.

For fax-only applications not requiring IVR functions, use the universal port configuration files and number of ports-per-board guidelines in this section.

| Maximum universal (fax and IVR) ports per board |       |              | Configuration file for... |                          |
|---|-------|--------------|---------------------------|--------------------------|
| Board   | Ports | Restrictions | mu-law                    | A-law                    |
| AG 2000/200                                     | 8     | None         | <i>agpi2fax.cfg</i>       | <i>agpi2faxa.cfg</i>     |
| AG 2000-BRI                                     | 8     | None         | <i>agpi2bri_fax.cfg</i>   | <i>agpi2bri_faxa.cfg</i> |
| AG 2000/400                                     | 16    | None         | <i>agpi2fax.cfg</i>       | <i>agpi2faxa.cfg</i>     |

The AG 2000/400 board provides DSPs for up to 16 ports of IVR and fax, but only eight line interfaces. Eight additional line interfaces must be provided with additional hardware.

## 6. Working with images

---

### Image format characteristics

This topic explains file storage formats and image encoding formats as specified in ITU T.4 and T.6. Use this information to choose the most effective combination of image format characteristics for your host system and your application.

NaturalFax uses TIFF-F or TIFF-S image files. TIFF-F files (tagged image file format for fax) and TIFF-S files (tagged image file format, profile S for T.37) use tags to specify the format of the graphics file. Refer to [T.37](#) and [TIFF-S](#) for more information about using TIFF-S image formats.

The image data in a TIFF-F or TIFF-S file has three basic characteristics:

- Encoding format (always 1D for TIFF-S)
- Resolution format (always LOW for TIFF-S)
- Page width (always A4 for TIFF-S)

### Encoding formats

NaturalFax supports four encoding formats:

| Encoding format   | Description  |
|-------------------|--|
| NFX_ENCODE_1D     | One-dimensional (MH) encoding  |
| NFX_ENCODE_2D     | Two-dimensional (MR) encoding  |
| NFX_ENCODE_MMR    | MMR (modified modified read) encoding  |
| NFX_ENCODE_TIFF_S | Encoding used for T.37 file formats (which sets TIFF-S encoding, resolution, and page width) |

A fax terminal can support any of the following combinations of encoding formats:

- 1D only
- 1D and 2D
- 1D, 2D, and MMR

Fax terminals are backwards-compatible: a fax terminal cannot support 2D or MMR encoding without also supporting 1D encoding.

All three encoding formats produce identical images. No information is lost during a conversion to a different encoding format. The final image appears identical to the original image.

1D, 2D, and MMR provide the same image quality, but differ in the following attributes:

| Attribute                              | Description   |
|--|---|
| Data compression                       | Affects how much disk space is required for file storage and how much transmission time is needed for image transfer.                   |
| CPU resource consumption for encoding  | Affects application performance and availability of CPU resources for other needs.  |
| Impact of errors                       | Includes whether a local error affects one line or multiple lines, and whether error correction mode (ECM) must be enabled.             |
| Compatibility with other fax terminals | Affects the probability that an application will need to convert an image to a different encoding in order to transmit it successfully. |

MMR encoding consumes the smallest amount of disk space and the least transmission time. An equivalent image in 2D encoding consumes more disk space and transmission time than MMR, and 1D encoding consumes more disk space and transmission time than 2D.

Encoding and decoding an MMR image consumes more CPU resources than encoding and decoding a 2D image. 1D images consume the least CPU resources for encoding and decoding.

A local error in a 1D image affects a single line in the document. An error in a 2D image may affect several subsequent lines as well. An error in an MMR image affects the remainder of the page. Therefore, MMR images can only be sent with error correction mode (ECM) enabled. 1D and 2D images can be sent with or without ECM enabled.

Files stored in MMR format may have to be converted to another encoding format before they can be transmitted to some fax machines. Use MMR to minimize disk storage requirements when the system has plenty of available CPU resources for format conversions.

**Note:** You can use MMR encoding to store fax image files and still transmit or receive faxes with ECM disabled by using on-the-fly conversion. To enable on-the-fly conversion, refer to [Performing online image conversion](#).

1D encoding enables files to be transmitted directly to any fax machine without conversion. Files stored in 1D format use the lowest common denominator of encoding formats. Use 1D encoding to conserve CPU resources and minimize format conversion needs.



The following table summarizes the advantages and disadvantages of each image encoding format. The image quality for all three encoding formats is identical.

| Encoding | Storage space | CPU resources | An error affects...                                       | Transmit time | ECM requirement |
|----------|---------------|---------------|---|---------------|-----------------|
| 1D       | Most          | Least         | A single line   | Most          | Optional        |
| 2D       | More          | More          | Multiple lines  | More          | Optional        |
| MMR      | Least         | Most          | Remainder of page, which forces retransmission of a frame | Least         | Mandatory       |

## Resolution formats

NaturalFax supports three resolution formats:

| Resolution format         | Description                    |
|---------------------------|--------------------------------|
| NFX_RESOLUTION_LOW        | 3.85 scan lines/mm, vertically |
| NFX_RESOLUTION_HIGH       | 7.7 scan lines/mm, vertically  |
| NFX_RESOLUTION_SUPER_HIGH | 15.4 scan lines/mm, vertically |

A fax terminal can support only LOW resolution, both LOW and HIGH resolution, or all three resolution values, LOW, HIGH, and SUPER\_HIGH. Converting an image from higher to lower resolution modifies the image data, and information is lost. The resulting image is more compact, which saves storage space and transmission time. When converting from higher to lower resolution, NaturalFax favors black pixels to prevent the loss of thin black lines.

NaturalFax can convert from lower to higher resolution by duplicating lines. However, conversion from lower to higher makes little sense since the additional redundant information does not improve image quality and requires more storage space and transmission time.

## Page width formats

NaturalFax supports the following page width formats:

| Page width format | Description                                  |
|-------------------|--|
| NFX_PAGE_WIDTH_A4 | 8.25 inches wide, 200 pixels/inch resolution |
| NFX_PAGE_WIDTH_B4 | 10 inches wide, 200 pixels/inch resolution   |
| NFX_PAGE_WIDTH_A3 | 11.9 inches wide, 200 pixels/inch resolution |

A fax terminal may support only A4, both A4 and B4, or all three page width values, A4, B4, and A3. Converting an image from a wider to a narrower page modifies the image data, and information is lost. The resulting image is more compact, which saves storage space and transmission time. NaturalFax shrinks lines to fit so that the image is not cut off. Black pixels are favored for shrinkage to prevent the loss of thin black lines.

NaturalFax can convert from a narrower page width to a wider page width by appending white space.

## Options for storing and converting image data

Each image stored in a TIFF-F file has a specific set of image format characteristics. TIFF-S files always store the image using 1D encoding, low resolution, and A4 page width. During image transfer, the image characteristics of the stored image may be different from the values of the image being sent or received. Image format conversion can occur on-the-fly during the fax session, or it can be initiated offline by the application.

Consider these factors when you design your application, select your TIFF-F file storage characteristics, and determine your conversion settings:

| Factor               | Description  |
|----------------------|--|
| Machine restrictions | Not all fax terminals can handle the full range of image characteristics. A more capable machine always supports the values of a less capable machine. At the start of a fax session, the receiver announces its capabilities (DIS frame). The sender chooses the image format for the transmission (DCS frame) within the limits of the receiver's capabilities. If the file to be sent has an image format that is beyond the capabilities of the receiver, the sender must perform on-the-fly image conversion. |
| File size            | Some image formats consume less data storage space than others, but may require more CPU resources from the host system for encoding. You may want to use a particular format for all transactions if data storage space and transmission time (and telephone charges) are primary considerations for application design.  |
| CPU resources        | Converting between one image format and another requires host processor execution time. The execution time required for conversions is the limiting factor if the system's CPU is heavily loaded by database or other operations in addition to fax functions. The application may accept a longer transmission time in return for conserving CPU resources. Offline conversion of an image from one format to another can be performed to help minimize the execution load.                                       |
| Bad line replacement | NaturalFax performs bad line analysis by decoding a line and comparing its length in pixels to the line length appropriate to the page width in use. During image conversion, bad lines may be eliminated or replaced, as determined by the value of the badlineaction parameter.  |

| Factor                 | Description  |
|------------------------|--|
| Document compatibility | NaturalFax transmits files that are in TIFF-F or TIFF-S format. The TIFF-F files may have different image attributes on different pages. NaturalFax performs on-the-fly conversions to ensure that mixed image files are transmitted as a single document. |

## T.37 and TIFF-S

T.37 describes an end-to-end fax session, using e-mail as the transport. The ITU-T document for a T.37 session completely describes the format and addressing conventions required to transport a fax using e-mail protocols as the transport over the internet to a G3 device. Part of the ITU-T defines TIFF-S, the file format used for fax over e-mail.

TIFF-S is a subset of the commonly used TIFF-F file format. TIFF-S has the following page characteristics across all pages:

- 1D encoding
- LOW resolution
- A4 page width

TIFF-S also specifies the physical layout of the TIFF file. No optional TIFF fields are allowed when writing TIFF tag data, to ensure that minimal black and white fax transmission occurs.

## 7. Function summary

---

### Document queue functions

The following functions control NaturalFax document queues:

| Function                        | Synchronous/<br>Asynchronous | Description  |
|---------------------------------|------------------------------|--|
| <a href="#">nfxCreateQueue</a>  | Synchronous                  | Creates a queue for either sending or receiving documents. |
| <a href="#">nfxDestroyQueue</a> | Synchronous                  | Deletes the specified document queue.                      |
| <a href="#">nfxEnqueueDoc</a>   | Synchronous                  | Adds a document to a document queue.                       |
| <a href="#">nfxResetQueue</a>   | Synchronous                  | Resets all documents in a queue to unsent.                 |

### Transmit and receive documents functions

The following functions control fax operations:

| Function                         | Synchronous/<br>Asynchronous | Description  |
|----------------------------------|------------------------------|--|
| <a href="#">nfxSendFax</a>       | Asynchronous                 | Starts negotiating with the remote fax receiver and subsequently sends all the documents in the send queue.  |
| <a href="#">nfxReceiveFax</a>    | Asynchronous                 | Starts negotiating as the called fax terminal and then places all received documents into the receive queue. |
| <a href="#">nfxAnswerFaxPoll</a> | Asynchronous                 | Provides a queue of documents to send in response to a poll request from the remote sending fax terminal.    |
| <a href="#">nfxStopSession</a>   | Asynchronous                 | Stops a fax send or receive operation in progress.   |

## Image format conversion functions

The following functions verify or convert the image formats of document files:

| Function                             | Synchronous/<br>Asynchronous | Description   |
|--------------------------------------|------------------------------|---|
| <a href="#">nfxCheckTIFF</a>         | Synchronous                  | Verifies that the specified file is in TIFF-F or TIFF-S format. |
| <a href="#">nfxConvertFileDirect</a> | Synchronous                  | Performs file format conversion.                                |

## Managing pages and document contents functions

The following functions move pages of a document from one file to another:

| Function                     | Synchronous/<br>Asynchronous | Description  |
|------------------------------|------------------------------|--|
| <a href="#">nfxMergeFile</a> | Synchronous                  | Combines multiple TIFF-F or TIFF-S files into a single TIFF-F or TIFF-S file.            |
| <a href="#">nfxSplitFile</a> | Synchronous                  | Splits a single TIFF-F or TIFF-S file into a specified number of TIFF-F or TIFF-S files. |

## Status monitoring functions

The following functions retrieve NaturalFax status information:

| Function                            | Synchronous/<br>Asynchronous | Description  |
|-------------------------------------|------------------------------|--|
| <a href="#">nfxGetSessionStatus</a> | Synchronous                  | Returns status information during a fax send or receive operation. |
| <a href="#">nfxGetDocStatus</a>     | Synchronous                  | Returns the status of a document in the specified queue.           |

## 8. Function reference

---

### Using the function reference

This section provides a comprehensive, alphabetically-ordered reference to the NaturalFax functions. A prototype of each function is shown with the function description and details of all arguments and return values. A typical function description includes:

|                      |  |
|----------------------|--|
| <b>Prototype</b>     | <p>The prototype is shown followed by a list of the function arguments. Data types include:</p> <ul style="list-style-type: none"><li>• WORD 16-bit unsigned</li><li>• DWORD 32-bit unsigned</li><li>• INT16 16-bit signed</li><li>• INT32 32-bit signed</li><li>• BYTE 8-bit unsigned</li></ul> <p>If a function argument is a data structure, the complete data structure is defined. Refer to the Overview of NaturalFax data structures for a description of all data structures and parameters.</p> |
| <b>Return values</b> | <p>The return value for a function is either SUCCESS or an error code. For asynchronous functions, a return value of SUCCESS indicates the function was initiated; subsequent events indicate the status of the operation.</p> <p>Refer to the Alphabetical error summary for a list of all errors returned by NaturalFax functions.</p>   |
| <b>Events</b>        | <p>If events are listed, the function is asynchronous and is complete when the DONE event is returned. If there are no events listed, the function is synchronous.</p> <p>Additional information such as reason codes and return values may be provided in the value field of the event.</p> <p>Refer to the Overview of events for information about all NaturalFax events and reason codes.</p>  |

## nfxAnswerFaxPoll

Provides a queue of documents to send in response to a poll request from the remote fax terminal.

### Prototype

DWORD **nfxAnswerFaxPoll** ( CTAHD *ctahd*, NFX\_QUEUE\_HANDLE *send\_queue\_handle*, NFX\_TRANSMIT\_PARMS *\*ptr\_transmit\_parms*)

| Argument                  | Description   |
|---------------------------|---|
| <i>ctahd</i>              | Context handle returned by <b>ctaCreateContext</b> .  |
| <i>send_queue_handle</i>  | Handle for queue of documents to send, returned by <b>nfxCreateQueue</b> .  |
| <i>ptr_transmit_parms</i> | <p>Pointer to NFX_TRANSMIT_PARMS structure, (or NULL to use default values), as follows:</p> <pre> typedef struct {     DWORD size;     DWORD modemtype; /* NFX_MODEM_TYPE_V17, NFX_MODEM_TYPE_V27, */                     /* or NFX_MODEM_TYPE_V29 */     DWORD minrate; /* NFX_BIT_RATE_2400, NFX_BIT_RATE_4800, */                     /* NFX_BIT_RATE_7200, NFX_BIT_RATE_9600, */                     /* NFX_BIT_RATE_12000, NFX_BIT_RATE_14400 */     DWORD resolution; /* NFX_RESOLUTION_HIGH, NFX_RESOLUTION_LOW, */                      /* or NFX_RESOLUTION_SUPER_HIGH */     DWORD encoding; /* NFX_ENCODE_1D, NFX_ENCODE_2D, or */                    /* NFX_ENCODE_MMR */     DWORD pagewidth; /* NFX_PAGE_WIDTH_A4, NFX_PAGE_WIDTH_B4, */                      /* or NFX_PAGE_WIDTH_A3 */     DWORD OTFmode; /* NFX_OTF_NEVER, NFX_OTF_ALWAYS, or */                   /* NFX_OTF_ONLY_IF_FAIL */     DWORD useECM; /* NFX_YES or NFX_NO */     DWORD useCNG; /* NFX_YES or NFX_NO */     DWORD PRIenabled; /* NFX_YES or NFX_NO */     DWORD timeout; /* number of seconds to wait for receiver */     DWORD retrainaction; /* NFX_RTN_REPEAT_PAGE or */                        /* NFX_RTN_NEXT_PAGE */     DWORD addheader; /* NFX_YES, NFX_NO or NFX_CUSTOM */     INT32 level; /* Transmit level in tenths of dBm */                /* (-150 to -60) */     INT32 threshold; /* Lowest level for receive, in tenths of dBm */     DWORD NSFlength; /* Length of NSF field or 0 if none */     char SID[NFX_MAX_SID]; /* Subscriber ID string */     BYTE NSF[NFX_MAX_NSF]; /* Default NSF for session */     char custom_header[NFX_MAX_HEADER]; /* Customizable fax header */     char SUB[NFX_MAX_SUB]; /* Sub-Address string */     DWORD useSUBADD; /* NFX_YES or NFX_NO */     DWORD txrate; /* NFX_BIT_RATE_2400, NFX_BIT_RATE_4800, */                  /* NFX_BIT_RATE_7200, NFX_BIT_RATE_9600, */                  /* NFX_BIT_RATE_12000, NFX_BIT_RATE_14400 */     DWORD ForceRate; /* NFX_YES or NFX_NO */ } NFX_TRANSMIT_PARMS; </pre> <p>See NFX_TRANSMIT_PARMS for complete field descriptions.</p> |

**Return values**

| Return value               | Description   |
|----------------------------|---|
| SUCCESS                    |   |
| CTAERR_FUNCTION_NOT_ACTIVE | The receive fax operation is not running on the specified context.                      |
| CTAERR_INVALID_CTAHD       | The specified context handle is invalid.  |
| CTAERR_INVALID_HANDLE      | The specified document queue handle is invalid.   |
| NFXERR_CONVERSION_REQUIRED | Current fax operation requires image conversion, but on-the-fly conversion is disabled. |
| NFXERR_QUEUE_EMPTY         | The specified document queue is empty.  |

**Events**

| Event                      | Description  |
|----------------------------|--|
| NFXEVN_CANNOT_OPEN_FILE    | <b>nfxAnswerFaxPoll</b> could not find the specified file in the document queue. The fax operation continues with the next document in the queue.  |
| NFXEVN_DOC_BEGIN           | Document transmission started.   |
| NFXEVN_DOC_END             | The transmitting fax terminal indicated that the last page of the document was sent. The value field contains SUCCESS or an error code.  |
| NFXEVN_PAGE_BEGIN          | Page transmission started.   |
| NFXEVN_PAGE_END            | The value field contains SUCCESS or an error code. SUCCESS indicates that the receiving fax terminal received the transmitted page, and sent an acknowledgment to the transmitting fax terminal. |
| NFXEVN_PROCEDURE_INTERRUPT | The called fax machine sent a procedure interrupt signal.  |
| NFXEVN_SEND_STARTED        | The polled fax session started.  |



| Event               | Description   |
|---------------------|---|
| NFXEVN_SESSION_DONE | <p>The fax session completed. The value field in this event may contain any of the following reason codes or an error code:</p> <ul style="list-style-type: none"> <li>CTA_REASON_FINISHED<br/>The fax function terminated normally.</li> <li>CTA_REASON_RELEASED<br/>The fax session ended because the call was disconnected.</li> <li>CTA_REASON_STOPPED<br/>The fax operation was cancelled by a call to <b>nfxStopSession</b>.</li> </ul> |

### Details

When NFXEVN\_POLLED is received, the application can call **nfxStopSession** to refuse the poll request from the called fax terminal.

**Note:** If **nfxAnswerPoll** is not called within three seconds of receiving NFXEVN\_POLLED, the fax session times out and the application receives an NFXEVN\_SESSION\_DONE event.

Receiving any of these events indicates that the fax session status was updated. Use **nfxGetSessionStatus** to examine the current session status in more detail when any of the information events are received.

### Using fax headers

Use the addheader parameter in the NFX\_TRANSMIT\_PARMS structure to add headers to the transmitted image data:

| Header option | Description   |
|---------------|---|
| NFX_NO        | Do not add a header.  |
| NFX_YES       | <p>Add a header which is a fixed string in the following format:</p> <pre>FROM transmit_SID Date Time Page N of M</pre> |
| NFX_CUSTOM    | Add a customized header of up to 80 ASCII characters.   |

Use C style format strings to include the page number, the date, and the time for each page:

| Format string | Description                        |
|---------------|------------------------------------|
| %d            | Date/time string.                  |
| %p            | Page number.                       |
| %P            | Total number of pages in this fax. |

| Format string | Description                                      |
|---------------|--|
| %y            | Current page number in fax session.              |
| %Y            | Total number of pages in all documents in queue. |

For example, the following code:


```
txparms.addheader = NFX_CUSTOM;
strcpy (txparms.custom_header, " Page %p of %P.\n
This is a test of custom headers %d");
```

Creates a fax header similar to the following example:

```
Page 1 of 5. This is a test of custom headers Fri Apr29 12:25:12 1999
```

Do not exceed the 80 character limit when using the custom header. Include expanded format strings (%d, %p, %P, %y, %Y) in the 80 character limit.

By default, addheader is set to NFX\_YES, and a header is added to the document being transmitted.

**Warning:**  Headers are required in certain countries, including those under FCC or DoC jurisdiction. Check telecommunications regulations in the target countries for your fax application.

You must define the environment variable NFXHEADERFONT to add a header. Refer to Setting environment variables in UNIX and Setting environment variables in Windows for more information.

**See also**

[nfxGetSessionStatus](#), [nfxReceiveFax](#)

## nfxCheckTIFF

Verifies that the specified file is in TIFF-F format, and analyzes the attributes of the TIFF-F or TIFF-S file.

### Prototype

DWORD **nfxCheckTIFF** ( CTAHD *ctahd*, char *\*input\_file\_name*, DWORD *\*number\_of\_pages*, DWORD *tracked\_pages*, NFX\_CHECK\_STATUS *\*page\_array*)

| Argument               | Description  |
|------------------------|--|
| <i>ctahd</i>           | Context handle returned by <b>ctaCreateContext</b> .                         |
| <i>input_file_name</i> | Pointer to the name of the file to check for TIFF-F or TIFF-S format.        |
| <i>number_of_pages</i> | Pointer to the location that receives the number of pages in the input file. |
| <i>tracked_pages</i>   | Number of pages to be reported in the array of NFX_CHECK_STATUS structures.  |

| Argument                 | Description  |
|--------------------------|--|
| <b><i>page_array</i></b> | <p>Pointer to an array of NFX_CHECK_STATUS structures. The count of entries in the array equals the value for <b><i>tracked_pages</i></b>. The NFX_CHECK_STATUS structure is defined as follows:</p> <pre>typedef struct {     DWORD size;           /* size of this structure          */     DWORD resolution;     /* NFX_RESOLUTION_HIGH,          */                         /* NFX_RESOLUTION_LOW, or        */                         /* NFX_RESOLUTION_SUPER_HIGH     */     DWORD encoding;       /* NFX_ENCODE_1D, NFX_ENCODE_2D, */                         /* NFX_ENCODE_MMR, or NFX_ENCODE_TIFF_S */     DWORD pagewidth;      /* NFX_PAGE_WIDTH_A4, NFX_PAGE_WIDTH_B4, */                         /* or NFX_PAGE_WIDTH_A3          */     DWORD lines;          /* Number of lines on the page    */     DWORD badlines;       /* Number of bad lines on the page */ } NFX_CHECK_STATUS;</pre> |

### Return values

| Return value             | Description  |
|--------------------------|--|
| SUCCESS                  | No bad lines were found.   |
| CTAERR_BAD_ARGUMENT      | Invalid function argument passed.  |
| CTAERR_INVALID_CTAHD     | The specified context handle is invalid.   |
| NFXERR_BAD_FILE_FORMAT   | The specified file is not in TIFF-F or TIFF-S format.  |
| NFXERR_CHECK_BAD_LINES   | The specified file has at least one bad line, but can still be transmitted with NaturalFax.  |
| NFXERR_CHECK_DIFF_ATTRIB | All pages in the specified file do not have the same attributes. The file may not be transmitted by NaturalFax, but the status information returned by <b>nfxCheckTIFF</b> is valid. |

### Events

None.

### Details

NaturalFax can transmit files in TIFF-F and TIFF-S format. **nfxCheckTIFF** reads and analyzes a specified file. It then stores the attributes of each page in a separate NFX\_CHECK\_STATUS structure in a returned array of NFX\_CHECK\_STATUS structures. The function fills in one structure per page until there are no more structures. If a pointer to ***page\_array*** is not passed to **nfxCheckTIFF**, the information is not stored. If the number of pages in the document is greater than the value of ***tracked\_pages***, the additional page information is not stored, but is still used to determine the return value for the function.

If the file analyzed by **nfxCheckTIFF** is a TIFF-S file, the reported attributes are 1D encoding, LOW resolution, and A4 page width. TIFF-S is not explicitly stated, and TIFF-S compliance is not verified.

If there is a problem in the TIFF-F file, **nfxCheckTIFF** returns NFXERR\_CHECK\_BAD\_LINES, but still produces a valid NFX\_CHECK\_STATUS structure array. Use this array to examine the page attributes to find any bad lines. The application can call **nfxConvertFileDirect** to remove the bad lines.

NaturalFax can transmit a file with bad lines.

When **nfxCheckTIFF** encounters a problem before it fills the array, it returns an error code other than NFXERR\_CHECK\_BAD\_LINES or NFXERR\_CHECK\_DIFF\_ATTRIB. In these cases, the information returned in the NFX\_CHECK\_STATUS structures may be incomplete.

**Note:** You must continue processing events during an active fax session. Make sure that file I/O intensive operations such as TIFF-F format verification do not interfere with the handling of events and cause the fax session to time out. The application should process events within three seconds.

Refer to Performing offline image conversion for more information.

#### See also

[\*\*nfxConvertFileDirect\*\*](#)

## nfxConvertFileDirect

Performs image format conversion on TIFF-F or TIFF-S files.

### Prototype

DWORD **nfxConvertFileDirect** ( CTAHD *ctahd*, char *\*input\_file*, char *\*output\_file*, NFX\_CONVERT\_PARMS *\*ptr\_convert\_parms*, DWORD *\*ptr\_converted\_pages*)

| Argument                   | Description  |
|----------------------------|--|
| <i>ctahd</i>               | Context handle returned by <b>ctaCreateContext</b> .   |
| <i>input_file</i>          | Pointer to an existing file to be converted.   |
| <i>output_file</i>         | Pointer to a file to be created to receive the conversion output.  |
| <i>ptr_convert_parms</i>   | <p>Pointer to an <a href="#">NFX_CONVERT_PARMS</a> structure identifying the output format. The structure is defined as follows:</p> <pre>typedef struct {     DWORD size;     DWORD type;           /* NFX_TIFF_F or NFX_TIFF_S          */     DWORD resolution;     /* NFX_RESOLUTION_HIGH,            */                           /* NFX_RESOLUTION_LOW,            */                           /* or NFX_RESOLUTION_SUPER_HIGH    */     DWORD encoding;       /* NFX_ENCODE_1D, NFX_ENCODE_2D,   */                           /* NFX_ENCODE_MMR, or NFX_ENCODE_TIFF_S */     DWORD pagewidth;      /* NFX_PAGE_WIDTH_A4, NFX_PAGE_WIDTH_B4 */                           /* or NFX_PAGE_WIDTH_A3            */     DWORD badlineaction;  /* NFX_BAD_LINE_ACTION_NONE,       */                           /* NFX_BAD_LINE_ACTION_DROP,       */                           /* NFX_BAD_LINE_ACTION_REPT, or    */                           /* NFX_BAD_LINE_ACTION_TICK        */ } NFX_CONVERT_PARMS;</pre> <p>See <a href="#">NFX_CONVERT_PARMS</a> for complete field descriptions.</p> |
| <i>ptr_converted_pages</i> | Pointer to a location to receive a count of pages converted.   |

### Return values

| Return value             | Description                              |
|--------------------------|--|
| SUCCESS                  |  |
| CTAERR_BAD_ARGUMENT      | Invalid function argument passed.        |
| CTAERR_FILE_NOT_FOUND    | The specified input file does not exist. |
| CTAERR_FILE_OPEN_FAILURE | Error opening file.                      |
| CTAERR_FILE_READ_FAILURE | Error reading file.                      |

| Return value              | Description   |
|---------------------------|---|
| CTAERR_FILE_WRITE_FAILURE | Error writing to file.  |
| CTAERR_INVALID_CTAHD      | The specified context handle is invalid.                          |
| NFXERR_BAD_FILE_FORMAT    | The specified input file is not in a valid format for conversion. |

## Events

None.

## Details

**nfxConvertFileDirect** modifies the encoding, resolution, and page width of the input file to match the attributes specified in the NFX\_CONVERT\_PARMS structure.

**nfxConvertFileDirect** also removes bad lines (as specified by the badlineaction parameter) from a TIFF-F or TIFF-S file. **nfxConvertFileDirect** can accept an input file that has different attributes on different pages and produce an output file that has the same attributes on all pages.

Specifying encoding with NFX\_ENCODE\_TIFF\_S or NFX\_TIFF\_S overrides all parameters to the TIFF-S format. Encoding is 1D, resolution is LOW, and pagewidth is A4, regardless of any other values specified in this structure.

**nfxConvertFileDirect** operates only on TIFF-F or TIFF-S files. Output files are always in TIFF-F or TIFF-S format, and always have the same attributes on all pages.

This function is synchronous and consumes significant processor execution time. Pending events in this thread can be delayed by many seconds until this function completes. Errors occurring during the conversion process are indicated in the function return value. The number of pages converted successfully is given in **ptr\_converted\_pages**.

**Note:** Continue processing events during an active fax session. Make sure that file I/O intensive operations such as format conversion do not interfere with the handling of events and cause the fax session to time out. The application should process events within three seconds.

Refer to Performing offline image conversion for more information.

## See also

[nfxCheckTIFF](#)

## nfxCreateQueue

Creates a queue for either sending or receiving fax documents.

## Prototype

DWORD **nfxCreateQueue** ( CTAHD *ctahd*, int *queue\_type*, NFX\_QUEUE\_HANDLE *\*ptr\_queue\_handle*)

| Argument     | Description  |
|--------------|--|
| <i>ctahd</i> | Context handle returned by <b>ctaCreateContext</b> . |

| Argument                | Description  |
|-------------------------|--|
| <i>queue_type</i>       | Type of document queue (either NFX_DOC_RECEIVE or NFX_DOC_TRANSMIT). |
| <i>ptr_queue_handle</i> | Pointer to a location to receive a valid queue handle.               |

### Return values

| Return value             | Description   |
|--------------------------|---|
| SUCCESS                  |   |
| CTAERR_BAD_ARGUMENT      | Invalid function argument passed.                   |
| CTAERR_INVALID_CTAHD     | The specified context handle is invalid.            |
| CTAERR_INVALID_HANDLE    | The specified document queue handle is invalid.     |
| CTAERR_OUT_OF_MEMORY     | Memory allocation failed.                           |
| NFXERR_OPEN_QUEUE_FAILED | An error occurred while accessing a document queue. |

### Events

None.

### Details

The document queue data structure is maintained internally by NaturalFax and returns a handle to use with all other NaturalFax functions. Each document queue is associated with a specified context. Subsequent calls to **nfxEnqueueDoc**, **nfxDestroyQueue**, **nfxSendFax**, **nfxReceiveFax**, or **nfxAnswerFaxPoll** verify that the document queue is associated with the same context as the current operation.

You can associate more than one document queue with a given context, but you can associate only one send queue and one receive queue with an active fax session. For example, to send multiple queues, you call **nfxSendFax** for each queue.

Refer to Using document queues for more information.

### See also

[nfxDestroyQueue](#), [nfxEnqueueDoc](#)

## nfxDestroyQueue

Deletes the specified document queue and frees the corresponding queue handle.

### Prototype

DWORD **nfxDestroyQueue** ( CTAHD *ctahd*, NFX\_QUEUE\_HANDLE *queue\_handle*)

| Argument            | Description  |
|---------------------|--|
| <i>ctahd</i>        | Context handle returned by <b>ctaCreateContext</b> . |
| <i>queue_handle</i> | Identifies the queue to be freed.                    |

#### Return values

| Return value          | Description                                     |
|-----------------------|---|
| SUCCESS               |   |
| CTAERR_BAD_ARGUMENT   | Invalid function argument passed.               |
| CTAERR_INVALID_CTAHD  | The specified context handle is invalid.        |
| CTAERR_INVALID_HANDLE | The specified document queue handle is invalid. |

#### Events

None.

#### Details

This function deallocates the structures that maintain a document queue and deletes any remaining document entries in the designated document queue. After the application invokes **nfxDestroyQueue**, the specified queue handle is no longer valid for subsequent calls. The files identified by the queue structures are untouched.

The queue structure is deallocated also when **ctaCloseServices** is called.

Refer to Terminating and shutting down for more information.

#### See also

[nfxCreateQueue](#), [nfxEnqueueDoc](#)



## nfxEnqueueDoc

Allocates a document entry in a document queue.

### Prototype

DWORD **nfxEnqueueDoc** ( CTAHD *ctahd*, NFX\_QUEUE\_HANDLE *queue\_handle*, char *\*file\_name*, NFX\_DOC\_PARMS *\*ptr\_doc\_parms*, DWORD *\*ptr\_doc\_number*)

| Argument              | Description   |
|-----------------------|---|
| <i>ctahd</i>          | Context handle returned by <b>ctaCreateContext</b> .  |
| <i>queue_handle</i>   | Identifies a valid queue to receive the new document.   |
| <i>file_name</i>      | Pointer to an existing file to send or a file to create.  |
| <i>ptr_doc_parms</i>  | <p>Pointer to an NFX_DOC_PARMS structure (or NULL to use default values), as follows:</p> <pre>typedef struct {     DWORD size;           /* Size of this structure */     DWORD resolution;     /* NFX_RESOLUTION_HIGH, NFX_RESOLUTION_LOW */                         /* or NFX_RESOLUTION_SUPER_HIGH */     DWORD encoding;       /* NFX_ENCODE_1D, NFX_ENCODE_2D or */                         /* NFX_ENCODE_MMR */     DWORD pagewidth;      /* NFX_PAGE_WIDTH_A4, NFX_PAGE_WIDTH_B4 or */                         /* NFX_PAGE_WIDTH_A3 */ } NFX_DOC_PARMS;</pre> <p>See NFX_DOC_PARMS for complete field descriptions.</p> |
| <i>ptr_doc_number</i> | Pointer to a location to receive a valid document number.   |

### Return values

| Return value           | Description  |
|------------------------|--|
| SUCCESS                |  |
| CTAERR_BAD_ARGUMENT    | Invalid function argument passed.  |
| CTAERR_FILE_EXISTS     | Attempted to enqueue an existing file to a receive queue. No queue entry is made.  |
| CTAERR_FILE_NOT_FOUND  | Attempted to enqueue a nonexistent file to the send queue. No queue entry is made. |
| CTAERR_INVALID_CTAHD   | The specified context handle is invalid.   |
| CTAERR_INVALID_HANDLE  | The specified document queue handle is invalid.                                    |
| NFXERR_BAD_FILE_FORMAT | The specified file is not in TIFF-F or TIFF-S format.                              |

| Return value          | Description   |
|-----------------------|---|
| NFXERR_QUEUE_TOO_LATE | Attempted to enqueue a file in an active transmit or receive queue too late in the fax session. No queue entry is made. |

## Events

None.

## Details

Use **nfxEnqueueDoc** to build transmit queues or receive queues of documents before calling **nfxSendFax**, **nfxReceiveFax**, or **nfxAnswerFaxPoll**. You can also use **nfxEnqueueDoc** to add documents to transmit queues or receive queues while a send or receive operation is under way.

The **file\_name** is passed as a separate argument and differs with each call to **nfxEnqueueDoc**. The file specified by **file\_name** enqueued to a transmit queue must exist in the correct format for transmission (TIFF-F or TIFF-S). The file specified by **file\_name** enqueued to a receive queue must not exist when it is enqueued to the receive queue.

The document number is placed in **ptr\_doc\_number** and identifies the position of the document in the queue. The document number and the queue handle are used for subsequent calls to **nfxGetDocStatus**.

The parameter values in NFX\_DOC\_PARMS can be changed with each call to **nfxEnqueueDoc**, but most of them are likely to be set once and used thereafter without changes.

Refer to Using document queues for more information.

## See also

[nfxCreateQueue](#), [nfxDestroyQueue](#)

## nfxGetDocStatus

Returns status of a document in the specified queue.

## Prototype

DWORD **nfxGetDocStatus** ( CTAHD *ctahd*, NFX\_QUEUE\_HANDLE *queue\_handle*, DWORD *doc\_number*, NFX\_DOC\_STATUS \**ptr\_doc\_status*)

| Argument            | Description   |
|---------------------|---|
| <i>ctahd</i>        | Context handle returned by <b>ctaCreateContext</b> .      |
| <i>queue_handle</i> | Identifies the queue that holds the specified document.   |
| <i>doc_number</i>   | Number of the document to retrieve status information on. |

| Argument                     | Description   |
|------------------------------|---|
| <b><i>ptr_doc_status</i></b> | <p>Pointer to an NFX_DOC_STATUS structure (as follows) to receive the document status:</p> <pre>typedef struct {     DWORD size;     DWORD docnum;           /* Entry number in queue          */     DWORD processedstatus; /* Has this document been sent?   */                            /* NFX_NO, NFX_YES, or error      */     DWORD lasterror;       /* Last error code (if any) for doc */     DWORD pagecount;       /* Number of pages (for receive)   */     DWORD startpage;       /* Page to start at (for retransmit) */     DWORD lastpagesent;    /* Last page # successfully sent   */     DWORD retranscount;    /* Number of retries               */     DWORD docencoding;     /* 1D (MH) or 2D (MR) or MMR      */     DWORD docresolution;   /* Resolution: low, high, superhigh */     DWORD docwidth;        /* Width: A4, A3, B4              */     DWORD negotiatedencoding; /* Encoding used for transfer    */     DWORD negotiatedrate;   /* Baud rate used for transfer     */     DWORD negotiatedresolution; /* Resolution used for transfer */     DWORD negotiatedwidth;  /* Page width used for transfer    */     DWORD badlinecount;    /* Number of bad lines in doc rec'ved; */                            /* or, in ECM, bad frames txed/rxed */     DWORD doctime;         /* Time document processing started */     DWORD duration;        /* Seconds elapsed processing doc   */     char filename[NFX_FILENAME_MAX]; /* filename arg from          */                            /* nfxEnqueueDoc               */     DWORD modemtype;       /* Modem used to transfer document */ } NFX_DOC_STATUS;</pre> |

### Return values

| Return value          | Description                                     |
|-----------------------|---|
| SUCCESS               |   |
| CTAERR_BAD_ARGUMENT   | A function argument is invalid.                 |
| CTAERR_INVALID_CTAHD  | The specified context handle is invalid.        |
| CTAERR_INVALID_HANDLE | The specified document queue handle is invalid. |

### Events

None.

### Details

The NFX\_DOC\_STATUS structure is updated after a document is processed (NFXEVN\_DOC\_END). The processedstatus field indicates whether the document was processed. If the processedstatus field is set, the other fields in this structure contain information on the status of the document.

If the **nfxStopSession** function is called in the middle of a fax session, the value of the processedstatus field for the current document is zero.

The lasterror field in the NFX\_DOC\_STATUS structure can contain any of the CTAERR or NFXERR values. For more information, refer to the Alphabetical error summary.

**See also**[nfxEnqueueDoc](#)**nfxGetSessionStatus**

Returns status information during a fax send or receive operation.

**Prototype**DWORD **nfxGetSessionStatus** ( CTAHD *ctahd*, NFX\_FAX\_STATUS *\*ptr\_fax\_status*)

| Argument              | Description   |
|-----------------------|---|
| <i>ctahd</i>          | Context handle returned by <b>ctaCreateContext</b> .  |
| <i>ptr_fax_status</i> | Pointer to an NFX_FAX_STATUS structure, as follows: <pre>typedef struct {     DWORD size;     DWORD rate;     DWORD ecm;          /* Error correction mode: NFX_YES or NFX_NO    */     DWORD resolution;     DWORD encoding;     DWORD pagewidth;     DWORD mode;         /* NFX_MODE_IDLE, NFX_MODE_NEGOTIATING,          */                         /* NFX_MODE_TRANSMITTING, NFX_MODE_RECEIVING,    */                         /* NFX_MODE_DISCONNECTING, NFX_MODE_FINISHED    */     DWORD error;        /* Last error code generated                      */     DWORD docnumber;    /* Current document in progress                  */     DWORD pagenumber;   /* Current page of the document                 */     DWORD badlines;     /* Number of bad lines (or bad frames in       */                         /* ECM mode) during current session            */     char filename[NFX_FILENAME_MAX]; /* Name of current file */                         /* processed                                   */     char remoteSID[NFX_MAX_SID];                         /* Received SID from remote station            */     BYTE remoteNSF[NFX_MAX_NSF];                         /* Received NSF from remote station           */     DWORD snr;          /* Signal to noise ratio in dB                  */     DWORD rx_training_zeros;                         /* Training zeros in the TCF                   */                         /* in tens of milliseconds                    */     char remoteSUB[NFX_MAX_SUB];                         /* Received Sub-Address string from          */                         /* remote station                           */     DWORD sub_sent;                         /* Set if Sub-Address frame is sent to       */                         /* remote station                          */     DWORD modemtype;    /* Modem used to transfer document             */ } NFX_FAX_STATUS;</pre> |

**Return values**

| Return value         | Description                              |
|----------------------|--|
| SUCCESS              |  |
| CTAERR_BAD_ARGUMENT  | A function argument is invalid.          |
| CTAERR_INVALID_CTAHD | The specified context handle is invalid. |

## Events

None.

## Details

Use **nfxGetSessionStatus** to return additional information about the state of the fax session when the most recent event occurred.

The error field in the NFX\_FAX\_STATUS structure may contain any of the CTAERR or NFXERR values. For more information, refer to the Alphabetical error summary.

**Note:** Modem metrics are currently supported on the AG and CG boards.

For more information, refer to Monitoring fax session status.

## See also

[nfxReceiveFax](#), [nfxSendFax](#)

## nfxMergeFile

Combines multiple TIFF-F or TIFF-S files into a single TIFF-F or TIFF-S file.

## Prototype

DWORD **nfxMergeFile** ( CTAHD *ctahd*, char *output\_file\_name*, DWORD *number\_input\_files*, char *\*input\_files*[], DWORD *\*number\_of\_pages*)

| Argument                  | Description  |
|---------------------------|--|
| <i>ctahd</i>              | Context handle returned by <b>ctaCreateContext</b> .     |
| <i>output_file_name</i>   | Name of output file.                                     |
| <i>number_input_files</i> | Number of input file names being passed to the function. |
| <i>input_files</i>        | Pointer to an array of input file names.                 |
| <i>number_of_pages</i>    | Pointer to the number of pages moved to the output file. |

## Return values

| Return value             | Description                              |
|--------------------------|--|
| SUCCESS                  |  |
| CTAERR_BAD_ARGUMENT      | A function argument is invalid.          |
| CTAERR_FILE_NOT_FOUND    | The specified input file does not exist. |
| CTAERR_FILE_OPEN_FAILURE | Error opening file.                      |
| CTAERR_FILE_READ_FAILURE | Error reading file.                      |

| Return value              | Description   |
|---------------------------|---|
| CTAERR_FILE_WRITE_FAILURE | Error writing to file.  |
| CTAERR_INVALID_CTAHD      | The specified context handle is invalid.                            |
| NFXERR_BAD_FILE_FORMAT    | One of the specified input files is not in TIFF-F or TIFF-S format. |

## Events

None.

## Details

Although TIFF-F files can contain pages with different attributes, **nfxMergeFile** copies whole pages from one file to another. It does not convert or examine attributes. All pages in the input files are copied sequentially into a single file named **output\_file\_name**. At least one **input\_file** file must exist. The contents of the input files are not modified.

The resulting TIFF-F or TIFF-S output file can have different attributes on different pages.

Use **nfxCheckTIFF** to examine the attributes of all input files before operating on them with either **nfxMergeFile** or **nfxSplitFile**.

**Note:** Make sure the application continues processing events during an active fax session. If file I/O intensive operations such as merging files interfere with the handling of events, the fax session could time out. The application should process events within three seconds.

Refer to Performing offline image conversion for more information.

## See also

[nfxCheckTIFF](#), [nfxSplitFile](#)

## nfxReceiveFax

Starts the receive side of a T.30 protocol fax session and stores all received document files in the specified receive queue. These files must not exist before you use this function.

### Prototype

DWORD **nfxReceiveFax** (CTAHD *ctahd*, NFX\_QUEUE\_HANDLE *receive\_queue\_handle*, NFX\_RECEIVE\_PARMS *\*ptr\_receive\_parms*)

| Argument                    | Description  |
|-----------------------------|--|
| <i>ctahd</i>                | Context handle returned by <b>ctaCreateContext</b> .   |
| <i>receive_queue_handle</i> | Handle for queue of documents to receive, returned by <b>nfxCreateQueue</b> .  |
| <i>ptr_receive_parms</i>    | <p>Pointer to an <a href="#">NFX_RECEIVE_PARMS</a> structure (NULL to use default values), as follows:</p> <pre> typedef struct {     DWORD size;     DWORD modemtype;      /* NFX_MODEM_TYPE_V17, NFX_MODEM_TYPE_V27, */                         /* or NFX_MODEM_TYPE_V29 */     DWORD minrate;        /* NFX_BIT_RATE_2400, NFX_BIT_RATE_4800, */                         /* NFX_BIT_RATE_7200, NFX_BIT_RATE_9600, */                         /* NFX_BIT_RATE_12000, NFX_BIT_RATE_14400 */     DWORD resolution;     /* NFX_RESOLUTION_HIGH, */                         /* NFX_RESOLUTION_LOW, or */                         /* NFX_RESOLUTION_SUPER_HIGH */     DWORD encoding;        /* NFX_ENCODE_1D, NFX_ENCODE_2D, */                         /* NFX_ENCODE_MMR, or NFX_ENCODE_TIFF_S */     DWORD pollingenabled; /* NFX_YES or NFX_NO */     DWORD badlineaction;  /* NFX_BAD_LINE_ACTION_NONE, */                         /* NFX_BAD_LINE_ACTION_DROP, */                         /* NFX_BAD_LINE_ACTION_REPT, or */                         /* NFX_BAD_LINE_ACTION_TICK */     DWORD pagewidth;      /* NFX_PAGE_WIDTH_A4, NFX_PAGE_WIDTH_B4, */                         /* or NFX_PAGE_WIDTH_A3 */     DWORD OTFmode;        /* NFX_OTF_NEVER, NFX_OTF_ALWAYS, or */                         /* NFX_OTF_ONLY_IF_FAIL */     DWORD useECM;          /* NFX_YES or NFX_NO */     DWORD lineerrors;      /* % line errors before retrain negative */     INT32 level;           /* Tx level in tenths of dBm(-150 to -60) */     INT32 threshold;      /* Lowest lev. for receive, tenths of dBm */     DWORD NSFlength;      /* Length of NSF field or 0 if none */     char SID[NFX_MAX_SID]; /* Subscriber ID string */     BYTE NSF[NFX_MAX_NSF]; /* Default NSF for session */     char SUB[NFX_MAX_SUB]; /* Sub-Address string */     DWORD useSUBADD;      /* NFX_YES or NFX_NO */ }NFX_RECEIVE_PARMS; </pre> <p>See <a href="#">NFX_RECEIVE_PARMS</a> for complete field descriptions.</p> |

### Return values

| Return value | Description |
|--------------|-------------|
| SUCCESS      |             |

| Return value           | Description  |
|------------------------|--|
| CTAERR_BAD_ARGUMENT    | A function argument is invalid.                        |
| CTAERR_FUNCTION_ACTIVE | A fax function is already active on the given context. |
| CTAERR_INVALID_CTAHD   | The specified context handle is invalid.               |
| CTAERR_INVALID_HANDLE  | The specified document queue handle is invalid.        |

## Events

| Event                      | Description  |
|----------------------------|--|
| NFXEVN_CANNOT_OPEN_FILE    | <b>nfxReceiveFax</b> could not create the specified file. The fax operation continues with the next document in the queue.   |
| NFXEVN_DOC_BEGIN           | A start-of-message signal was received.  |
| NFXEVN_DOC_END             | An end-of-message signal was received. The event value field can contain any of the following values or an error code: <ul style="list-style-type: none"> <li>• NFX_T30_EOM<br/>An end-of-message signal was received from the called fax terminal (more documents to follow). If no more documents are in the receive queue, NaturalFax puts any additional sent documents into the last document enqueued. This may result in a TIFF-F file with different attributes on some pages.</li> <li>• NFX_T30_EOP<br/>An end-of-procedure signal was received from the called fax terminal (no more documents to follow).</li> </ul> |
| NFXEVN_PAGE_BEGIN          | A start-of-page signal was received.   |
| NFXEVN_PAGE_END            | The value field contains SUCCESS or an error code. SUCCESS indicates that the receiving fax terminal received an end-of-page signal, and sent an acknowledgment to the transmitting fax terminal.  |
| NFXEVN_POLLED              | The calling fax machine polled the application.  |
| NFXEVN_PROCEDURE_INTERRUPT | A procedure interrupt was received from the called fax machine.  |



| Event                    | Description  |
|--------------------------|--|
| NFXEVN_RECEIVE_STARTED   | Confirms that the fax session has started.   |
| NFXEVN_REMOTE_IDENTIFIED | The remote terminal was identified. You can now verify the remote SID and NSF by checking the session status.  |
| NFXEVN_SESSION_DONE      | <p>The fax session completed with one of the following terminating reasons, or an error code:</p> <ul style="list-style-type: none"> <li>CTA_REASON_FINISHED<br/>The fax session terminated normally.</li> <li>CTA_REASON_RELEASED<br/>The fax session ended because the call was disconnected.</li> <li>CTA_REASON_STOPPED<br/>The fax session was cancelled by a call to <b>nfxStopSession</b>.</li> </ul> |

### Details

Using an encoding value of NFX\_ENCODE\_TIFF\_S forces the values for encoding, resolution, and page width to be overridden. The new values for the TIFF-S are 1D, LOW and A4, respectively.

NFXEVN\_\*\*\* events indicate the progress of the fax session and the completion of the fax session.

**Note:** You must continue processing events during an active fax session. Make sure that file I/O intensive operations do not interfere with the handling of events and cause the fax session to time out. The application should process events within three seconds.

Some computer-based fax programs can send documents with different image attributes in a single fax session. For example, a PC-based fax transmitter can send a cover page at low resolution and send the rest of the document at high resolution. NaturalFax handles this situation differently depending on the number of documents enqueued. If the application enqueued multiple documents, separate documents are created for each change in resolution. If only one document remains in the queue, the pages with different resolution will be added to the last document in the queue.

Receipt of any of the previously listed events indicates that the fax session status was updated. Use **nfxGetSessionStatus** to examine the current session status in more detail.

After a fax session completes, the application releases the call and tears down or resets any document queues. NaturalFax sends a DCN (disconnect) frame at the end of a fax session to signal the remote fax terminal to disconnect. The application can use functions from the NCC service to release the call after a completed fax session.

A normal fax session ends with the transmitter sending the DCN (disconnect) frame, signaling the remote fax terminal to disconnect. Some fax terminals may disconnect the call before they receive or transmit the DCN frame. Under these conditions, the NaturalFax application may receive an NCCEVN\_CALL\_DISCONNECTED event before it receives the NFXEVN\_SESSION\_DONE event.

The application should wait for the NFXEVN\_SESSION\_DONE event, and then release the call as usual. The NFXEVN\_SESSION\_DONE event will contain a reason code of CTA\_REASON\_RELEASED or CTA\_REASON\_FINISHED.

To enable polling, make sure that the called fax terminal has the pollingenabled parameter in NFX\_RECEIVE\_PARMs set to NFX\_YES. If polling is enabled and the sender subsequently requests polling, the event NFXEVN\_POLLED is received. Use **nfxAnswerFaxPoll** to continue the fax operation.

For more information, refer to Receiving faxes.

**See also**

[nfxAnswerFaxPoll](#), [nfxSendFax](#)

## nfxResetQueue

Resets all the flags in a queue to mark all documents as unsent.

### Prototype

DWORD **nfxResetQueue** ( CTAHD *ctahd*, NFX\_QUEUE\_HANDLE *queue\_handle*)

| Argument            | Description  |
|---------------------|--|
| <i>ctahd</i>        | Context handle returned by <b>ctaCreateContext</b> . |
| <i>queue_handle</i> | Identifies the queue to reset.                       |

### Return values

| Return value          | Description                                     |
|-----------------------|---|
| SUCCESS               |   |
| CTAERR_BAD_ARGUMENT   | A function argument is invalid.                 |
| CTAERR_INVALID_CTAHD  | The specified context handle is invalid.        |
| CTAERR_INVALID_HANDLE | The specified document queue handle is invalid. |

### Events

None.

### Details

Each document in a queue is flagged as sent when it is successfully sent. If a problem occurs during transmission, the whole queue can be resubmitted and only unflagged documents are sent.

Calling **nfxResetQueue** resets all flags so that all documents in the specified queue can be resent. This function is typically used in broadcast applications that send the same queue of documents on multiple phone calls.

If this function is called with a receive queue, it also changes the queue type to a transmit queue. This use is typical of a fax store-and-forward application.

Refer to Resetting a document queue for more information.

### See also

[nfxCreateQueue](#), [nfxDestroyQueue](#), [nfxEnqueueDoc](#)

## nfxSendFax

Starts the transmit side of a T.30 protocol fax session, and sends all the documents in the send queue to the called fax terminal.

**Prototype**

DWORD **nfxSendFax** ( CTAHD *ctahd*, NFX\_QUEUE\_HANDLE *send\_queue\_handle*, NFX\_TRANSMIT\_PARMS *\*ptr\_transmit\_parms*, NFX\_QUEUE\_HANDLE *receive\_queue\_handle*, NFX\_RECEIVE\_PARMS *\*ptr\_receive\_parms*)

| Argument                    | Description  |
|-----------------------------|--|
| <i>ctahd</i>                | Context handle returned by <b>ctaCreateContext</b> .   |
| <i>send_queue_handle</i>    | Handle of a queue of documents to send, returned by <b>nfxCreateQueue</b> .  |
| <i>ptr_transmit_parms</i>   | <p>Pointer to an <a href="#">NFX_TRANSMIT_PARMS</a> structure (or NULL to use default values), as follows:</p> <pre> typedef struct {     DWORD size;     DWORD modemtype; /* NFX_MODEM_TYPE_V17, NFX_MODEM_TYPE_V27,    */                       /* or NFX_MODEM_TYPE_V29                      */     DWORD minrate;  /* NFX_BIT_RATE_2400, NFX_BIT_RATE_4800,    */                       /* NFX_BIT_RATE_7200, NFX_BIT_RATE_9600,    */                       /* NFX_BIT_RATE_12000, NFX_BIT_RATE_14400   */     DWORD resolution; /* NFX_RESOLUTION_HIGH, NFX_RESOLUTION_LOW or */                       /* NFX_RESOLUTION_SUPER_HIGH                */     DWORD encoding;  /* NFX_ENCODE_1D, NFX_ENCODE_2D, or         */                       /* NFX_ENCODE_MMR                          */     DWORD pagewidth; /* NFX_PAGE_WIDTH_A4, NFX_PAGE_WIDTH_B4, or */                       /* NFX_PAGE_WIDTH_A3                       */     DWORD OTFmode;   /* NFX_OTF_NEVER, NFX_OTF_ALWAYS,          */                       /* or NFX_OTF_ONLY_IF_FAIL                 */     DWORD useECM;    /* NFX_YES or NFX_NO                       */     DWORD useCNG;    /* NFX_YES or NFX_NO                       */     DWORD PRIenabled; /* NFX_YES or NFX_NO                       */     DWORD timeout;   /* Number of seconds to wait for receiver   */     DWORD retrainaction; /* NFX_RTN_REPEAT_PAGE or                 */                       /* NFX_RTN_NEXT_PAGE                      */     DWORD addheader; /* NFX_YES, NFX_NO or NFX_CUSTOM            */     INT32 level;     /* Transmit level in tenths of dBm          */                       /* (-150 to -60)                          */     INT32 threshold; /* Lowest lev. for receive, in tenths of dBm */     DWORD NSFlength; /* Length of NSF field or 0 if none         */     char SID[NFX_MAX_SID]; /* Subscriber ID string                    */     BYTE NSF[NFX_MAX_NSF]; /* Default NSF for session                 */     char custom_header[NFX_MAX_HEADER];                       /* Customizable fax header                 */     char SUB[NFX_MAX_SUB]; /* Sub-Address string                      */     DWORD useSUBADD; /* NFX_YES or NFX_NO                       */     DWORD txrate;    /* NFX_BIT_RATE_2400, NFX_BIT_RATE_4800,    */                       /* NFX_BIT_RATE_7200, NFX_BIT_RATE_9600,    */                       /* NFX_BIT_RATE_12000, NFX_BIT_RATE_14400   */     DWORD ForceRate; /* NFX_YES or NFX_NO                       */ } NFX_TRANSMIT_PARMS; </pre> <p>See <a href="#">NFX_TRANSMIT_PARMS</a> for complete field descriptions.</p> |
| <i>receive_queue_handle</i> | Handle of a queue to receive polled documents (NULL to disable polling).   |

| Argument                        | Description  |
|---------------------------------|--|
| <b><i>ptr_receive_parms</i></b> | <p>Pointer to an <a href="#">NFX_RECEIVE_PARMS</a> structure (or NULL to use default values), as follows:</p> <pre>typedef struct {     DWORD size;     DWORD modemtype; /* NFX_MODEM_TYPE_V17, NFX_MODEM_TYPE_V27, */                     /* or NFX_MODEM_TYPE_V29 */     DWORD minrate; /* NFX_BIT_RATE_2400, NFX_BIT_RATE_4800, */                   /* NFX_BIT_RATE_7200, NFX_BIT_RATE_9600, */                   /* NFX_BIT_RATE_12000, NFX_BIT_RATE_14400 */     DWORD resolution; /* NFX_RESOLUTION_HIGH, NFX_RESOLUTION_LOW, */                      /* or NFX_RESOLUTION_SUPER_HIGH */     DWORD encoding; /* NFX_ENCODE_1D, NFX_ENCODE_2D, */                    /* NFX_ENCODE_MMR, or NFX_ENCODE_TIFF_S */     DWORD pollingenabled; /* NFX_YES or NFX_NO */     DWORD badlineaction; /* NFX_BAD_LINE_ACTION_NONE, */                        /* NFX_BAD_LINE_ACTION_DROP, */                        /* NFX_BAD_LINE_ACTION_REPT, or */                        /* NFX_BAD_LINE_ACTION_TICK */     DWORD pagewidth; /* NFX_PAGE_WIDTH_A4, NFX_PAGE_WIDTH_B4, or */                     /* NFX_PAGE_WIDTH_A3 */     DWORD OTFmode; /* NFX_OTF_NEVER, NFX_OTF_ALWAYS, or */                   /* NFX_OTF_ONLY_IF_FAIL */     DWORD useECM; /* NFX_YES or NFX_NO */     DWORD lineerrors; /* % Line errors before retrain negative */     INT32 level; /* Xmit level in tenths of dBm (-150 to -60) */     INT32 threshold; /* Lowest lev. for receive, in tenths of dBm */     DWORD NSFlength; /* Length of NSF field or 0 if none */     char SID[NFX_MAX_SID]; /* Subscriber ID string */     BYTE NSF[NFX_MAX_NSF]; /* Default NSF for session */     char SUB[NFX_MAX_SUB]; /* Sub-Address string */     DWORD useSUBADD; /* NFX_YES or NFX_NO */ } NFX_RECEIVE_PARMS;</pre> <p>See <a href="#">NFX_RECEIVE_PARMS</a> for complete field descriptions.</p> |

### Return values

| Return value           | Description   |
|------------------------|---|
| SUCCESS                |   |
| CTAERR_BAD_ARGUMENT    | A function argument is invalid.                               |
| CTAERR_FUNCTION_ACTIVE | A fax function is already active on the given context handle. |
| CTAERR_INVALID_CTAHD   | The specified context handle is invalid.                      |
| CTAERR_INVALID_HANDLE  | The specified document queue handle is invalid.               |
| NFXERR_QUEUE_EMPTY     | The specified document queue is empty.                        |

**Events**

| Event                      | Description   |
|----------------------------|---|
| NFXEVN_CANNOT_OPEN_FILE    | <b>nfxSendFax</b> could not find the specified file. The fax operation continues with the next document in the queue.   |
| NFXEVN_DOC_BEGIN           | A start-of-message signal was transmitted.  |
| NFXEVN_DOC_END             | An end-of-message signal was transmitted. The value field for this event contains SUCCESS or an error code.   |
| NFXEVN_PAGE_BEGIN          | A start-of-page signal was transmitted.   |
| NFXEVN_PAGE_END            | The value field contains SUCCESS or an error code. SUCCESS indicates that the receiving fax terminal received the transmitted page, and sent an acknowledgment to the transmitting fax terminal.  |
| NFXEVN_PROCEDURE_INTERRUPT | The called fax terminal acknowledged a procedure interrupt signal.  |
| NFXEVN_RECEIVE_STARTED     | The remote terminal acknowledged a poll request.  |
| NFXEVN_REMOTE_IDENTIFIED   | The remote terminal was identified. You can now verify the remote SID and NSF by checking the session status.   |
| NFXEVN_SEND_STARTED        | The fax session started.  |
| NFXEVN_SESSION_DONE        | <p>The fax session completed. The value field for this event can contain any of the following reasons or an error code:</p> <ul style="list-style-type: none"> <li>• CTA_REASON_FINISHED<br/>The fax function terminated normally.</li> <li>• CTA_REASON_RELEASED<br/>The fax session ended because the call was disconnected.</li> <li>• CTA_REASON_STOPPED<br/>The fax session was cancelled by a call to <b>nfxStopSession</b>.</li> </ul> |

## Details

The specified context must have an established call in the conversation state.

Events indicate the progress and completion of the fax session. The previously listed events indicate that the fax session status was updated. The application can use **nfxGetSessionStatus** to examine the current session status in more detail when any of the information events is received.

**Note:** The application must continue processing events during an active fax session. If file I/O intensive operations interfere with the handling of events, the fax session could time out. The application should process events within three seconds.

TIFF-F files with different image attributes on different pages can be sent using **nfxSendFax**.

**nfxSendFax** uses the NFX\_TRANSMIT\_PARMS and NFX\_RECEIVE\_PARMS structures to control fax transmission or fax reception behavior. It sets context values first using NFX\_RECEIVE\_PARMS and then with NFX\_TRANSMIT\_PARMS. Some common context values can be set using both parameters (for example, SID-Subscriber Identification). NaturalFax sets these common values with the values specified in NFX\_TRANSMIT\_PARMS by overwriting values set using NFX\_RECEIVE\_PARMS.

## Using fax headers

Use the addheader parameter in the NFX\_TRANSMIT\_PARMS structure to add headers to the transmitted image data:

| Header option | Description   |
|---------------|---|
| NFX_NO        | Do not add a header.  |
| NFX_YES       | Add a header that is a fixed string in the following format:<br>FROM transmit_SID Date Time Page N of M |
| NFX_CUSTOM    | Add a customized header of up to 80 ASCII characters.   |

Use C style format strings to include the page number, the date, and the time in the custom header:

| Format string | Description                                      |
|---------------|--|
| %d            | Date/time string.                                |
| %p            | Page number.                                     |
| %P            | Total number of pages in this fax.               |
| %y            | Current page number in fax session.              |
| %Y            | Total number of pages in all documents in queue. |

For example, the following code:

```
txparms.addheader = NFX_CUSTOM;
strcpy (txparms.custom_header, " Page %p of %P.\n
This is a test of custom headers %d");
```

Creates a fax header similar to the following example:

```
Page 1 of 5. This is a test of custom headers Fri Apr29
12:25:12 1999
```

Do not exceed the 80 character limit when using the custom header. Include expanded format strings (%d, %p, %P, %y, %Y) in the 80 character limit.

By default, addheader is set to NFX\_YES, and a header is added to the document being transmitted.

**Warning:**



Headers are mandated in certain countries, including those under FCC or DoC jurisdiction. Check telecommunications regulations in the target countries for your fax application.

You must define the environment variable NFXHEADERFONT to add a header. Refer to Setting environment variables in UNIX and Setting environment variables in Windows for more information.

## Using fax polling

In addition to sending documents, **nfxSendFax** supports polling. The fax polling operation allows the called fax terminal to respond by sending a queue of documents to the calling fax terminal. The calling fax terminal sends its send queue before trying to poll the called fax terminal. If the **send\_queue\_handle** is NULL, **nfxSendFax** tries to poll the called fax terminal. The event NFXEVN\_RECEIVE\_STARTED indicates that the remote fax terminal responded to the poll request and the calling fax terminal began receiving the polled documents.

**Note:** Either the **send\_queue\_handle** or the **receive\_queue\_handle** can be NULL, but not both.

Specifying NFX\_ENCODE\_TIFF\_S in the NFX\_RECEIVE\_PARMS structure overrides the encoding, resolution, and pagewidth values to the values for TIFF-S. The new values are 1D, LOW and A4, respectively.

## Sending a procedure interrupt

Use **nfxSendFax** to transmit a procedure interrupt (PRI) signal to the called terminal after completing the transmission of all documents in the send queue. A PRI signal ends an **nfxSendFax** operation and prevents the receiving fax terminal from hanging up the call. The receiving fax terminal prompts a person to pick up the handset, accepting the request and enabling the application to perform voice functions (such as playing a voice prompt, or recording a message). If no one picks up the handset or the receiving fax terminal does not respond to the PRI, the transmitting fax terminal completes operations normally.

To send a procedure interrupt, the application must set the PRIenabled parameter in NFX\_TRANSMIT\_PARMS to NFX\_YES. A procedure interrupt request is transmitted to the called fax terminal *after* the last document in the send queue is transmitted. The default setting for PRIenabled is NFX\_NO. When the called terminal acknowledges a procedure interrupt request, the event NFXEVN\_PROCEDURE\_INTERRUPT is sent to the application, followed by an NFXEVN\_SESSION\_DONE event. Voice functions may be performed after receiving the done event.

**Note:** Disable polling before using the procedure interrupt feature.



Refer to Transmitting faxes for more information. Refer to Polling the called fax terminal for more information about fax polling.

See also

[nfxGetSessionStatus](#), [nfxReceiveFax](#)

## nfxSplitFile

Splits a single TIFF-F or TIFF-S file into a specified number of TIFF-F or TIFF-S files.

### Prototype

DWORD **nfxSplitFile** ( CTAHD *ctahd*, char *input\_file\_name*, DWORD *number\_output\_files*, char *\*output\_files*[], DWORD *\*number\_of\_pages*)

| Argument                   | Description  |
|----------------------------|--|
| <i>ctahd</i>               | Context handle returned by <b>ctaCreateContext</b> . |
| <i>input_file_name</i>     | Name of input file.                                  |
| <i>number_output_files</i> | Number of output file names passed to function.      |
| <i>output_files</i>        | Pointer to an array of output file names.            |
| <i>number_of_pages</i>     | Pointer to the number of pages processed.            |

### Return values

| Return value              | Description   |
|---------------------------|---|
| SUCCESS                   |   |
| CTAERR_BAD_ARGUMENT       | A function argument is invalid.                                   |
| CTAERR_FILE_OPEN_FAILURE  | Error opening file.   |
| CTAERR_FILE_NOT_FOUND     | The specified input file does not exist.                          |
| CTAERR_FILE_READ_FAILURE  | Error reading file.   |
| CTAERR_FILE_WRITE_FAILURE | Error writing to file.  |
| CTAERR_INVALID_CTAHD      | The specified context handle is invalid.                          |
| NFXERR_BAD_FILE_FORMAT    | The specified input file is not in a format valid for conversion. |

### Events

None.

## Details

**nfxSplitFile** examines the TIFF-F or TIFF-S input file and creates multiple TIFF-F or TIFF-S output files. The pages of the input file are considered a string of pages. The files named in the **output\_files** array are written - one page to a file - until the last file is reached. All remaining pages are copied into the last file. If you specify only two output files, **nfxSplitFile** is a convenient way to remove cover pages from a received TIFF-F or TIFF-S file.

Although TIFF-F files can contain pages with different attributes, **nfxSplitFile** does not convert or examine attributes. It simply copies whole pages from a single input TIFF-F file to multiple TIFF-F output files.

Use **nfxCheckTIFF** to examine the attributes of all input files before operating on them with either **nfxMergeFile** or **nfxSplitFile**.

Compare **number\_output\_files** with **number\_of\_pages** for a quick indication of the number of output files created, and the number of pages in each output file. The number of files actually created is the smaller of **number\_of\_pages** and **number\_output\_files**. Extra output files are not used. There must be at least one output file.

**nfxSplitFile** does not modify the contents of the input file.

**Note:** You must continue processing events during an active fax session. Make sure that file I/O intensive operations, such as splitting files, do not interfere with the handling of events and cause the fax session to time out. The application should process events within three seconds.

Refer to Performing offline image conversion for more information.

## See also

[nfxCheckTIFF](#), [nfxMergeFile](#)

## nfxStopSession

Stops a current send or receive operation.

## Prototype

DWORD **nfxStopSession** ( CTAHD *ctahd*)

| Argument     | Description  |
|--------------|--|
| <i>ctahd</i> | Context handle returned by <b>ctaCreateContext</b> . |

## Return values

| Return value               | Description                                       |
|----------------------------|---|
| SUCCESS                    |   |
| CTAERR_BAD_ARGUMENT        | A function argument is invalid.                   |
| CTAERR_FUNCTION_NOT_ACTIVE | The specified context has no active fax function. |
| CTAERR_INVALID_CTAHD       | The specified context handle is invalid.          |

## Events

| Event               | Description   |
|---------------------|---|
| NFXEVN_SESSION_DONE | The fax session completed with the terminating reason code of CTA_REASON_STOPPED. |

## Details

After the application invokes **nfxStopSession**, NaturalFax immediately terminates the current send or receive operation. Cancellation is complete when the NFXEVN\_SESSION\_DONE event arrives.

Use this function when the application encounters an unexpected error. The fax session terminates abruptly, and is not completed with a T.30 DCN message. Use this function also when the telephone call is terminated by the remote end and the application is not using Natural Call Control or NMS ISDN for Natural Call Control (layer 4) functions for call control.

For example, when low layer ISDN protocols are used by the application, NaturalFax is not notified when the call is released. Therefore, the application must use **nfxStopSession** to terminate an active fax session. For more information, refer to Terminating and shutting down.

For more information on monitoring fax session status, refer to Monitoring fax session status.

## See also

[nfxAnswerFaxPoll](#), [nfxReceiveFax](#), [nfxSendFax](#)

## 9. Demonstration programs and utilities

---

### Summary of the demonstration programs and utilities

The following table provides a description of the NaturalFax demonstration programs and utilities:

| Program         | Description   |
|-----------------|---|
| <i>caller</i>   | Uses voice and fax functions to call a host system running <i>faxback</i> , then requests and receives a fax.   |
| <i>faxback</i>  | Uses voice and fax functions to receive a call, play a prompt menu, and respond to input by sending a fax to the caller.  |
| <i>nfxrecv</i>  | Receives a fax.   |
| <i>nfxsend</i>  | Sends a fax.  |
| <i>nfxcheck</i> | Verifies that the specified file is in TIFF-F or TIFF-S format. Retrieves the page format attributes of the file, the number of lines per page, and any bad line indications. |
| <i>nfxcnvrt</i> | Converts an input TIFF-F or TIFF-S file to an output TIFF-F or TIFF-S file with the specified attributes for encoding, resolution, page width, and bad line management.       |
| <i>nfxmerge</i> | Merges multiple TIFF-F or TIFF-S input files into a single TIFF-F or TIFF-S output file.  |
| <i>nfxsplit</i> | Splits a single TIFF-F or TIFF-S file into multiple TIFF-F or TIFF-S files.   |
| <i>nfxtxttf</i> | Converts an ASCII text file to a TIFF-F or TIFF-S file.   |

To display a list of arguments and their usage, invoke a NaturalFax demonstration program or utility from a command line without providing arguments. The list displays.

NaturalFax provides source code for *nfxsend*, *nfxrecv*, *faxback*, and *caller*.

Use *nfxsend* and *nfxrecv* for a quick demonstration of basic NaturalFax image transfer capabilities.

### Using voice and fax: caller

Uses voice and fax functions to call a host system running *faxback*, then requests and receives a fax.

#### Usage

```
caller [options]
```

*options* are:

| Option               | Description  | Default                |
|----------------------|--|------------------------|
| -b <i>n</i>          | Specifies the AG or CG board number <i>n</i> .   | 0                      |
| -q                   | Specifies that the board is a QX 2000 board.   | AG/CG board            |
| -s <i>n:m</i>        | Specifies the stream and timeslot.   | 0:0                    |
| -r <i>resolution</i> | Specifies the resolution (low, high, or superhigh).  | low                    |
| -e <i>encoding</i>   | Specifies the encoding (1D, 2D, MMR, or TIFF_S).   | 1D                     |
| -c <i>mode</i>       | Specifies the conversion mode:<br>y = yes (NFX_OTF_ALWAYS)<br>m = maybe (NFX_OTF_ONLY_IF_FAIL)<br>n = no (NFX_OTF_NEVER) | m                      |
| -p <i>protocol</i>   | Specifies the protocol (TCP name).   | nocc                   |
| -d <i>telno</i>      | Specifies the telephone number.  | No default.            |
| -E                   | Specifies that ECM mode is used when encoding is 1D or 2D.   | Do not use ECM.        |
| -v                   | Specifies verbose reporting of event information.  | Non-verbose reporting. |

## Description

Use *caller* to automate running *faxback*. *caller* performs the following tasks:

| Task | Description   |
|------|---|
| 1    | <i>caller</i> places a call to the host running <i>faxback</i> and requests a document by sending a touch tone.   |
| 2    | It receives the faxed document using NaturalFax, Natural Access, and ADI service functions. DTMF tones are used to synchronize <i>faxback</i> and <i>caller</i> . |
| 3    | <i>caller</i> begins by initializing the Natural Access environment.  |
| 4    | It opens the ADI service on a context associated with the specified board, stream, and timeslot.  |
| 5    | It places the call and sets up detection of the DTMF tone that signals the end of the prompt menu.  |

| Task | Description  |
|------|--|
| 6    | When it receives this tone, it generates the DTMF tone that corresponds to a request for a document catalog.   |
| 7    | <i>caller</i> then creates a document queue and enqueues nonexistent files that will store the received documents.   |
| 8    | <i>caller</i> sets receive parameters.   |
| 9    | It receives the document when <i>faxback</i> transmits it. Specifying TIFF-S encoding causes the calling application to receive a file using TIFF-S attributes of 1D encoding, LOW resolution, and A4 pagewidth. TIFF-S encoding overrides any values specified on the command line. |
| 10   | <i>caller</i> monitors all events until it receives the NFXEVN_SESSION_DONE event. It takes appropriate actions if the call is disconnected or if a board error occurs.  |
| 11   | After the fax is received, the document queue is destroyed and <i>caller</i> is set to place another call.   |

*caller* runs in an infinite loop. Press **Ctrl+C** to terminate the demonstration program.

## Using voice and fax: faxback

Uses voice and fax functions to receive a call, play a prompt menu, and respond to input by sending a fax to the caller. It can receive input from a live caller, or from the simulated caller provided by the *caller* demonstration program.

### Usage

```
faxback [options]
```

**options** are:

| Option               | Description  | Default     |
|----------------------|--|-------------|
| -b <i>n</i>          | Specifies the AG or CG board number <i>n</i> .   | 0           |
| -q                   | Specifies that the board is a QX 2000 board.   | AG/CG board |
| -s <i>n:m</i>        | Specifies the stream and timeslot.   | 0:0         |
| -r <i>resolution</i> | Specifies the resolution (low, high, or superhigh).  | low         |
| -e <i>encoding</i>   | Specifies the encoding (1D, 2D, or MMR).   | 1D          |
| -c <i>mode</i>       | Specifies the conversion mode:<br>y = yes (NFX_OTF_ALWAYS)<br>m = maybe (NFX_OTF_ONLY_IF_FAIL)<br>n = no (NFX_OTF_NEVER) | m           |

| Option             | Description  | Default                |
|--------------------|--|------------------------|
| -p <i>protocol</i> | Specifies the protocol (TCP name).                         | nocc                   |
| -E                 | Specifies that ECM mode is used when encoding is 1D or 2D. | Do not use ECM.        |
| -v                 | Specifies verbose reporting of event information.          | Non-verbose reporting. |

### Description

*faxback* demonstrates how to use voice, DTMF detection, and fax functions during a single telephone call. It uses API functions from NaturalFax, Natural Access, and the ADI service. *faxback* plays a fixed prompt menu to the caller, allows the caller to select a document, and faxes the document back to the caller.

*faxback* can be used interactively by calling it from a fax machine. It can also be used in conjunction with *caller*, a NaturalFax automated testing demonstration program.

### Running *faxback* on its own

*faxback* performs the following tasks:

| Task | Description  |
|------|--|
| 1    | <i>faxback</i> first initializes the Natural Access environment and then opens the ADI service on a context associated with the specified board, stream, and timeslot.   |
| 2    | When <i>faxback</i> receives a call, it plays a voice menu prompt requesting that the caller enter DTMF tones to select a document.                                      |
| 3    | <i>faxback</i> initializes a document queue and uses DTMF detection to determine which documents the caller selected.  |
| 4    | <i>faxback</i> enqueues the documents, sets transmit parameters, and transmits the requested documents.  |
| 5    | <i>faxback</i> monitors all events until it receives the NFXEVN_SESSION_DONE event. It takes appropriate actions if the call is disconnected or if a board error occurs. |
| 6    | After the fax is transmitted, the document queue is destroyed and <i>faxback</i> waits for the next incoming call.   |

*faxback* runs in an infinite loop. Press **Ctrl+C** to terminate the program.

### Running *faxback* and *caller* together

If *faxback* and *caller* are used together, the programs synchronize with each other using DTMF tones:

| Task | Description   |
|------|---|
| 1    | <i>faxback</i> initializes Natural Access and opens the ADI service on a context associated with the specified board, stream, and timeslot.                         |
| 2    | It places the call and sets up detection of the DTMF tone that signals the end of the prompt menu.  |
| 3    | When it receives this tone, it generates the DTMF tone that corresponds to a request for a document catalog.  |
| 4    | <i>caller</i> then creates a document queue, enqueues temporary files, sets receive parameters, and receives the document when <i>faxback</i> transmits it.         |
| 5    | <i>caller</i> monitors events until it receives the NFXEVN_SESSION_DONE event. It takes appropriate actions if the call is disconnected or if a board error occurs. |
| 6    | After the fax has been received, the document queue is destroyed, and <i>caller</i> places another call.  |

*caller* runs in an infinite loop. Press **Ctrl+C** to terminate the program.

## Receiving a fax: *nfxrecv*

Demonstrates receiving a fax. Use *nfxrecv* to receive a fax from a fax terminal, or from the *nfxsend* demonstration program.

### Usage

```
nfxrecv [options] filename1 [filename2 [ ... ]]
```

*options* are:

| Option               | Description  | Default     |
|----------------------|--|-------------|
| -b <i>n</i>          | Specifies the AG or CG board number <i>n</i> .   | 0           |
| -q                   | Specifies that the board is a QX 2000 board.   | AG/CG board |
| -s <i>n:m</i>        | Specifies the stream and timeslot.   | 0:0         |
| -r <i>resolution</i> | Specifies the resolution (low, high, or superhigh).  | low         |
| -e <i>encoding</i>   | Specifies the encoding (1D, 2D, MMR, or TIFF_S).   | 1D          |
| -c <i>mode</i>       | Specifies the conversion mode:<br>y = yes (NFX_OTF_ALWAYS)<br>m = maybe (NFX_OTF_ONLY_IF_FAIL)<br>n = no (NFX_OTF_NEVER) | m           |
| -p <i>protocol</i>   | Specifies the protocol (TCP name).   | nocc        |



| Option              | Description   | Default                |
|---------------------|---|------------------------|
| -w <i>pagewidth</i> | Specifies the page width (A4, B4, A3).                          | A4                     |
| -E                  | Specifies the using ECM mode is used when encoding is 1D or 2D. | Do not use ECM.        |
| -v                  | Specifies verbose reporting of event information.               | Non-verbose reporting. |

### Description

*nfxrecv* performs the following tasks:

| Task | Description  |
|------|--|
| 1    | <i>nfxrecv</i> receives a fax using NaturalFax and Natural Access functions.   |
| 2    | It initializes the Natural Access environment and opens the ADI service with the specified board, stream, and timeslot.  |
| 3    | <i>nfxrecv</i> then creates the fax document queue and enqueues the specified files.   |
| 4    | It answers the call, sets the receive parameters, and receives the fax. Specifying TIFF-S encoding causes the calling application to receive a file using TIFF-S attributes of 1D encoding, LOW resolution, and A4 page width. TIFF-S encoding overrides any values specified on the command line. |
| 5    | While waiting for the NFXEVN_SESSION_DONE event, <i>nfxrecv</i> monitors all events and takes appropriate action in case the call is disconnected or a board error occurs.   |
| 6    | After it receives the fax, <i>nfxrecv</i> releases the call, destroys the document queue and the event queue, and closes the context.  |

## Sending a fax: *nfxsend*

Demonstrates sending a fax. Use *nfxsend* to send a fax to a fax terminal or to the *nfxrecv* demonstration program.

### Usage

```
nfxsend [options] filename1 [filename2 [ ... ]]
```

*options* are:

| Option               | Description  | Default                |
|----------------------|--|------------------------|
| -b <i>n</i>          | Specifies the AG or CG board number <i>n</i> .   | 0                      |
| -q                   | Specifies that the board is a QX 2000 board.   | AG/CG board            |
| -s <i>n:m</i>        | Specifies the stream and timeslot.   | 0:0                    |
| -r <i>resolution</i> | Specifies the resolution (low, high, or superhigh).  | low                    |
| -e <i>encoding</i>   | Specifies the encoding (1D, 2D, or MMR).   | 1D                     |
| -c <i>mode</i>       | Specifies the conversion mode:<br>y = yes (NFX_OTF_ALWAYS)<br>m = maybe (NFX_OTF_ONLY_IF_FAIL)<br>n = no (NFX_OTF_NEVER) | m                      |
| -p <i>protocol</i>   | Specifies the protocol (TCP name).   | nocc                   |
| -d <i>telno</i>      | Specifies the telephone number.  | No default.            |
| -w <i>pagewidth</i>  | Specifies the page width (A4, B4, or A3).  | A4                     |
| -E                   | Specifies that ECM mode is used when encoding is 1D or 2D.   | Do not use ECM.        |
| -v                   | Specifies verbose reporting of event information.  | Non-verbose reporting. |
| -a <i>subadd</i>     | Specifies the sub-address.   | Not used.              |

| Option              | Description  | Default   |
|---------------------|--|-----------|
| -R <i>modemrate</i> | Specifies the modem type and modem rate:<br>Options are:<br>V17_14400<br>V17_12000<br>V17_9600<br>V17_7200<br>V29_9600<br>V29_7200<br>V27_4800<br>V27_2400 | V17_14400 |

### Description

*nfxsend* performs the following tasks:

| Task | Description  |
|------|--|
| 1    | <i>nfxsend</i> sends a fax using NaturalFax and Natural Access functions.  |
| 2    | It initializes the Natural Access environment and opens the ADI service with the specified board, stream, and timeslot.  |
| 3    | <i>nfxsend</i> creates the fax document queue and enqueues the specified files.  |
| 4    | It places the call, sets the transmit parameters, and sends the fax.   |
| 5    | While waiting for the NFXEVN_SESSION_DONE event, <i>nfxsend</i> monitors all events and takes appropriate action in case the call is disconnected or a board error occurs. |
| 6    | After the fax is sent, <i>nfxsend</i> releases the call, destroys the document queue and the event queue and closes the context.   |

## Verifying TIFF-F or TIFF-S format: *nfxcheck*

Determines if the input file is in TIFF-F or TIFF-S format. If the file is in TIFF-F or TIFF-S format, *nfxcheck* reports its page format attributes, number of lines per page, and bad line indications.

### Usage

```
nfxcheck input_file
```

### Description

If the input file is not a TIFF-F or TIFF-S file, *nfxcheck* displays an error message. Refer to the Alphabetical error summary for more information. If the input file is a TIFF-F file, *nfxcheck* displays the following information:

- Page attributes
- Number of lines per page
- Bad line indication

*nfxcheck* does not report the TIFF profile used to create the document. The utility cannot differentiate between a TIFF-F file and a TIFF-S file because a TIFF-F file can use the same page attributes for encoding, resolution, and pagewidth as a TIFF-S file.

*nfxcheck* does not alter or modify the input file.

## Converting TIFF-F or TIFF-S files: *nfxcnvrt*

Converts image format characteristics of an input TIFF-F or TIFF-S file, either according to default values or as specified in the command line, and saves the modified file under a designated new name.

### Usage

```
nfxcnvrt input_file output_file [convert_encoding [convert_resolution [convert_page_width [convert_bad_line]]]]
```

Unless you specify different attributes in the command line arguments, *nfxcnvrt* uses the default value. You can set any of the image format characteristics to any of their allowed values. The following table lists the image format characteristics along with their allowed values and their default values:

| Format characteristic     | Allowed values            | Default |
|---------------------------|---------------------------|---------|
| <i>convert_resolution</i> | HIGH<br>LOW<br>SUPER_HIGH | LOW     |
| <i>convert_encoding</i>   | 1D<br>2D<br>MMR<br>TIFF_S | 1D      |

| Format characteristic     | Allowed values               | Default |
|---------------------------|------------------------------|---------|
| <i>convert_page_width</i> | A4<br>B4<br>A3               | A4      |
| <i>convert_bad_line</i>   | DROP<br>REPT<br>TICK<br>NONE | REPT    |

### Description

The default attributes constitute image format capabilities that are found in every fax terminal. Any file processed by *nfxcnvrt* using its default values can be transmitted to, or received by, any fax terminal, regardless of its capabilities.

A screen message displays the number of pages processed when *nfxcnvrt* completes the conversion.

When you specify TIFF-S encoding, the utility converts the output file using TIFF-S attributes of 1D encoding, LOW resolution, and A4 pagewidth. TIFF-S encoding overrides any values specified on the command line.

To repair any bad lines in a file, run *nfxcnvrt* and specify an output format that is the same as the input format. This process removes bad lines by dropping the line, repeating the previous line, or replacing the line with a blank line and a tick mark in the margin, depending on the value selected for *convert\_bad\_line*. The original image cannot be reconstructed.

## Merging TIFF-F or TIFF-S files: *nfxmerge*

Merges multiple input TIFF-F or TIFF-S files into a single output TIFF-F or TIFF-S file.

### Usage

```
nfxmerge output_file input_file1 input_file2 [input_file3 [ ... ]]
```

### Description

*nfxmerge* copies several TIFF-F or TIFF-S document files into one output TIFF-F or TIFF-S file.

A screen message displays the number of pages processed when *nfxmerge* completes the operation.

*nfxmerge* does not alter the input files or modify the image format characteristics.

## Splitting TIFF-F or TIFF-S files: *nfxsplit*

Splits a single input TIFF-F or TIFF-S file into multiple output TIFF-F or TIFF-S files.

### Usage

```
nfxsplit input_file output_file1 output_file2 [output_file3 [ ... ]]
```

## Description

*nfxsplit* copies one page of the input file into each output file. If the input file contains more pages than there are named output files, *nfxsplit* copies all remaining pages into the last named output file. If the input file contains fewer pages than there are named output files, the extra named output files are not used.

*nfxsplit* does not alter the image format characteristics of the input file.

When *nfxsplit* completes the operation, a screen message displays the number of pages processed.

## Converting ASCII files: *nfxtxttf*

Converts an ASCII text file to a TIFF-F file.

## Usage

```
nfxtxttf input_file output_file [-s]
```

## Description

*nfxtxttf* converts an ASCII text file to a TIFF-F or TIFF-S file. The default output TIFF-F file has the following attributes: 1D encoding, HIGH resolution, and A4 page width.

When you specify the -s option, *nfxtxttf* creates a TIFF-S file. The TIFF-S file has 1D encoding, LOW resolution, and A4 page width.

Use this utility to prepare simple cover pages to fax.

The environment variable NMSTEXTFONT must be defined to run this utility. Refer to for Setting environment variables in UNIX and Setting environment variables in Windows more information.

## 10. Errors

---

### Alphabetical error summary

Error codes with their corresponding descriptions are presented in two tables for quick reference:

- Alphabetically ordered by error code name including a description of the problem
- Numerically ordered in hexadecimal with the decimal equivalent and error name

All Natural Access and Natural Access service functions return a status code. If the return code is not SUCCESS (0), it is an error code indicating that the function has failed and a reason for the failure.

An application can retrieve additional information about an error condition by calling [nfxGetSessionStatus](#) or [nfxGetDocStatus](#) and examining the error field in the NFX\_FAX\_STATUS structure or the lasterror field in NFX\_DOC\_STATUS. A table with these error codes is also provided.

NaturalFax error codes are prefixed with NFXERR, and are defined in *nfxdef.h*. For information about Natural Access error codes (prefixed with CTAERR), refer to the *Natural Access Developer's Reference Manual*.

The following table lists NaturalFax errors in alphabetical order. All errors are 32 bit.

| Error name             | Hex      | Decimal | Description  |
|------------------------|----------|---------|--|
| NFXERR_BAD_FILE_FORMAT | 0x050207 | 328199  | The specified file is not in TIFF-F or TIFF-S format.  |
| NFXERR_BAD_PAGE_SIZE   | 0x05020B | 328203  | Unsupported page size.   |
| NFXERR_BUFFER_UNDERRUN | 0x050211 | 328209  | On fax transmit, buffers of image data have not been provided. Check your host computer loading to be sure events and disk I/O are being processed in a timely fashion.  |
| NFXERR_CARRIER_LOST    | 0x05021D | 328221  | The received signal has fallen below the level specified by NFX_RECEIVE_PARMS.threshold, and the fax session has terminated.   |
| NFXERR_CHECK_BAD_LINES | 0x05020D | 328205  | Returned by <a href="#">nfxCheckTIFF</a> to indicate that there are bad lines in the TIFF-F or TIFF-S file. The file can still be transmitted by NaturalFax, and status information returned by <a href="#">nfxCheckTIFF</a> is valid. |

| Error name                   | Hex               | Decimal         | Description   |
|------------------------------|-------------------|-----------------|---|
| NFXERR_CHECK_DIFF_ATTRIB     | 0x05020E          | 328206          | Returned by <b>nfxCheckTIFF</b> to indicate that all pages in the TIFF-F or TIFF-S file do not have the same attributes (for encoding, page width, and resolution). The file may not be transmitted by NaturalFax, but the status information returned by <b>nfxCheckTIFF</b> is valid.                         |
| NFXERR_CONNECT_FAILED        | 0x050201          | 328193          | The called fax machine was not detected. Check that the number you are dialing is a fax machine.  |
| NFXERR_CONVERSION_REQUIRED   | 0x05020C          | 328204          | Cannot transmit file because there is a mismatch between the stored file format and the receiving fax terminal's capabilities and on-the-fly conversion has been disabled (otfmode in the receive or transmit parameters is set to NFX_OTF_NEVER).  |
| NFXERR_FAX_SERVICE           | 0x050219          | 328217          | A fatal error occurred in the FAX service.  |
| NFXERR_FXM_SERVICE           | 0x05021A          | 328218          | A fatal error occurred in the FXM service.  |
| NFXERR_INCOMPATIBLE_RECEIVER | 0x050216          | 328214          | The application has disabled on-the-fly conversion and the receiver cannot handle the specified image format.   |
| NFXERR_INTERNAL_ERROR        | 0x58000 - 0x5FFFF | 360448 - 393215 | An unexpected internal error has occurred.  |
| NFXERR_NEGOTIATION_FAILED    | 0x050202          | 328194          | Failed T.30 fax negotiation at the beginning of a session or between documents. When transmitting, this error indicates that training failed (usually due to bad phone line quality) or that the transmitter and receiver capabilities do not match. When receiving, this error indicates that training failed. |
| NFXERR_NO_MODEMS             | 0x050213          | 328211          | The required DSP files are not configured or loaded to the board. The FXM service failed to start.  |
| NFXERR_NO_PPM                | 0x050214          | 328212          | The receiver has not received a correct post-page message.  |
| NFXERR_NO_PPM_RESPONSE       | 0x050215          | 328213          | The receiver has not returned a correct response to the post-page message.  |
| NFXERR_NO_REMOTE             | 0x05021C          | 328220          | The called fax terminal did not send a DIS signal.  |



| Error name               | Hex      | Decimal | Description   |
|--------------------------|----------|---------|---|
| NFXERR_OPEN_QUEUE_FAILED | 0x05020A | 328202  | Could not open the specified document queue. Check that you have a valid transmit queue.  |
| NFXERR_PROTOCOL_ERROR    | 0x050217 | 328215  | A T.30 protocol error occurred.   |
| NFXERR_QUEUE_EMPTY       | 0x050206 | 328198  | The specified document queue is empty.  |
| NFXERR_QUEUE_TOO_LATE    | 0x050208 | 328200  | Attempted to enqueue a file in an active transmit queue too late in the fax session.  |
| NFXERR_RATE_TOO_LOW      | 0x050210 | 328208  | The receiving or transmitting fax terminal is attempting to receive or transmit at a rate lower than the minrate parameter.   |
| NFXERR_REMOTE_DCN        | 0x05021B | 328219  | The remote terminal has sent an unexpected disconnect (DCN) command frame.  |
| NFXERR_RETRAIN_NEGATIVE  | 0x05021E | 328222  | The page was received with errors, and NaturalFax has requested that the remote transmitter retransmit the last page.   |
| NFXERR_SESSION_FAILED    | 0x050209 | 328201  | The fax session failed due to an unexpected event. This error can indicate a T.30 timeout or failure to receive a post page message. Check your host processor CPU and I/O loading. |
| NFXERR_T5_TIMEOUT        | 0x05021F | 328223  | The receiver is not ready to receive after T5 (60 +/- 5 seconds).   |

## Numerical error summary

The following table lists NaturalFax errors in numerical order:

| Hex      | Decimal | Error name                |
|----------|---------|---------------------------|
| 0x050201 | 328193  | NFXERR_CONNECT_FAILED     |
| 0x050202 | 328194  | NFXERR_NEGOTIATION_FAILED |
| 0x050206 | 328198  | NFXERR_QUEUE_EMPTY        |
| 0x050207 | 328199  | NFXERR_BAD_FILE_FORMAT    |
| 0x050208 | 328200  | NFXERR_QUEUE_TOO_LATE     |

| Hex                    | Decimal            | Error name                 |
|------------------------|--------------------|----------------------------|
| 0x050209               | 328201             | NFXERR_SESSION_FAILED      |
| 0x05020A               | 328202             | NFXERR_OPEN_QUEUE_FAILED   |
| 0x05020B               | 328203             | NFXERR_BAD_PAGE_SIZE       |
| 0x05020C               | 328204             | NFXERR_CONVERSION_REQUIRED |
| 0x05020D               | 328205             | NFXERR_CHECK_BAD_LINES     |
| 0x05020E               | 328206             | NFXERR_CHECK_DIFF_ATTRIB   |
| 0x050210               | 328208             | NFXERR_RATE_TOO_LOW        |
| 0x050211               | 328209             | NFXERR_BUFFER_UNDERRUN     |
| 0x050212               | 328210             | NFXERR_BUFFER_OVERRUN      |
| 0x050213               | 328211             | NFXERR_NO_MODEMS           |
| 0x050214               | 328212             | NFXERR_NO_PPM              |
| 0x050215               | 328213             | NFXERR_NO_PPM_RESPONSE     |
| 0x050217               | 328215             | NFXERR_PROTOCOL_ERROR      |
| 0x050219               | 328217             | NFXERR_FAX_SERVICE         |
| 0x05021A               | 328218             | NFXERR_FXM_SERVICE         |
| 0x05021B               | 328219             | NFXERR_REMOTE_DCN          |
| 0x05021C               | 328220             | NFXERR_NO_REMOTE           |
| 0x05021D               | 328221             | NFXERR_CARRIER_LOST        |
| 0x05021E               | 328222             | NFXERR_RETRAIN_NEGATIVE    |
| 0x05021F               | 328223             | NFXERR_T5_TIMEOUT          |
| 0x058000 -<br>0x05FFFF | 360448 -<br>393215 | NFX_INTERNAL_ERROR         |

## 11. Events

### Overview of events

This section presents the three types of NaturalFax events:

- Completion
- Informational
- Confirmation

NaturalFax uses the same event reporting mechanism as Natural Access. The structure informs the application about which event occurred on which context, and includes information specific to the event.

All events in the Natural Access environment are represented by a C data structure, as shown in the following generic CTA\_EVENT:

```
typedef struct
{
    DWORD id;           /* Event id (LIBEVN_XXX in 'libdef.h') */
    CTAHD ctahd;        /* Context handle */
    DWORD timestamp;    /* Timestamp */
    DWORD userid;       /* Use-supplied id */
    DWORD size;         /* Size of buffer if buffer is not NULL */
    void *buffer;       /* Otherwise, may contain event-specific data */
    DWORD value;        /* Buffer pointer */
    UINTPTR objHd;      /* Event status or event-specific data */
    /* Service object handle */
} CTA_EVENT;
```

This structure, returned by **ctaWaitEvent**, informs the application which event occurred on which context, and provides additional information specific to the event.

The event structure contains the following fields:

| Field     | Description   |
|-----------|---|
| id        | Contains an event code defined in the library header file. All NaturalFax event codes are defined in the <i>nfdef.h</i> header file. All NaturalFax events are prefixed with NFXEVN_. All ADI service events are prefixed with ADIEVN_ and are defined in <i>adidef.h</i> . |
| ctahd     | Contains the context handle (returned from <b>ctaCreateContext</b> ).   |
| timestamp | Contains the time when the event was created.   |
| userid    | Contains the user-supplied ID. This field is unaltered by Natural Access and facilitates asynchronous programming. Its purpose is to correlate a context with an application object/context when events occur.  |
| size      | NaturalFax does not use this field.   |
| buffer    | NaturalFax does not use this field.   |

| Field | Description  |
|-------|--|
| value | This is an event-specific value. This field can hold a reason code or an error code. |
| objHd | Service object handle (for example, <i>swiHd</i> , <i>vcehd</i> , <i>callhd</i> ).   |

Three of the NaturalFax events, NFXEVN\_PAGE\_END, NFXEVN\_DOC\_END, and NFXEVN\_SESSION\_DONE, may contain reason or error codes in the value field, as specified in the following tables.

Refer to the *ADI Service Developer's Reference Manual* for a detailed explanation of events, errors, and reason codes in the ADI service. Refer to the *Natural Access Developer's Reference Manual* for a detailed explanation of events, errors, and reason codes in Natural Access.

## Completion events

The NFXEVN\_SESSION\_DONE event is sent as the last event for all asynchronous NaturalFax functions, regardless of the results or function requested. Always wait for this event before attempting the next function on the current context.

| Event               | Hex      | Description                            |
|---------------------|----------|--|
| NFXEVN_SESSION_DONE | 0x05200B | The current fax session has completed. |

## NFXEVN\_SESSION\_DONE values

The following values can appear in the value field of the NFXEVN\_SESSION\_DONE event:

| Error or reason         | Hex  | Description   |
|-------------------------|------|---|
| CTAERR_BAD_ARGUMENT     | 0x7  | Invalid modem rate specified for the modem type. Specify a valid modem type/modem rate combination.                             |
| CTAERR_FILE_EXISTS      | 0x24 | An attempt was made to create a file that already exists. Remove or rename the existing file, or specify a different file name. |
| CTAERR_FILE_NOT_FOUND   | 0x21 | The specified file does not exist. Create the file or specify a different file name.  |
| CTAERR_FILE_OPEN_FAILED | 0x25 | File open failed due to a system error. Verify that the file exists.  |

| Error or reason            | Hex  | Description  |
|----------------------------|------|--|
| CTAERR_FILE_READ_FAILED    | 0x27 | The file is not opened or is locked, or the expected amount of data could not be read. Make sure that the file is of the expected type. Verify that an incorrect handle was not used to close another file.  |
| CTAERR_FILE_WRITE_FAILED   | 0x28 | The file is not open or is locked, or the expected amount of data could not be written. Verify that an incorrect handle was not used to close another file.  |
| CTAERR_FUNCTION_ACTIVE     | 0xF  | A function is already running on the context. Be sure your application waits for a DONE event from other fax or voice operations before initiating a new fax operation.  |
| CTAERR_FUNCTION_NOT_ACTIVE | 0xE  | An attempt was made to stop or modify a function that was not running.   |
| CTAERR_FUNCTION_NOT_AVAIL  | 0xD  | A NaturalFax DSP file was not downloaded to the board.   |
| CTAERR_INVALID_CTAHD       | 0x5  | An invalid context handle was passed as an argument to a function or the context was destroyed by another thread.  |
| CTAERR_OUT_OF_MEMORY       | 0x6  | Unable to allocate memory for queue, driver or context, for play or record buffers, or for temporary storage. When this error occurs on a DONE event, it may mean that there was insufficient memory on the board.                                 |
| CTAERR_OUT_OF_RESOURCES    | 0x8  | A NaturalFax DSP function could not be started. Check to see that you are using the proper board configuration for your application. Check to see that you have not exceeded the maximum number of fax operations for your hardware configuration. |

| Error or reason            | Hex      | Description  |
|----------------------------|----------|--|
| CTA_REASON_FINISHED        | 0x1001   | The fax function terminated normally.  |
| CTA_REASON_RELEASED        | 0x1007   | The call was disconnected by the remote fax machine. Check that events are being handled in a timely fashion. Check your host processor CPU loading. The application will also receive ADIEVN_CALL_DISCONNECTED before NFXEVN_SESSION_DONE.          |
| CTA_REASON_STOPPED         | 0x1002   | The fax function was stopped by calling <a href="#">nfxStopSession</a> .   |
| NFXERR_BAD_FILE_FORMAT     | 0x050207 | The specified file is not in TIFF-F or TIFF-S format.  |
| NFXERR_BAD_PAGE_SIZE       | 0x05020B | Unsupported page size.   |
| NFXERR_BUFFER_UNDERRUN     | 0x050211 | On fax transmit, buffers of image data have not been provided. Check your host computer loading to be sure events and disk I/O are being processed in a timely fashion.  |
| NFXERR_CARRIER_LOST        | 0x05021D | The received signal has fallen below the level specified by NFX_RECEIVE_PARMS.threshold, and the fax session has been terminated.  |
| NFXERR_CONNECT_FAILED      | 0x050201 | The called fax machine was not detected. Check that the number you are dialing is connected to a fax machine.  |
| NFXERR_CONVERSION_REQUIRED | 0x05020C | Cannot transmit a file because there is a mismatch between the stored file format and the receiving fax terminal's capabilities and on-the-fly conversion has been disabled (otfmode in the receive or transmit parameters is set to NFX_OTF_NEVER). |
| NFXERR_FAX_SERVICE         | 0x050219 | A fatal error occurred in the FAX service.   |

| Error or reason              | Hex             | Description   |
|------------------------------|-----------------|---|
| NFXERR_FXM_SERVICE           | 0x05021A        | A fatal error occurred in the FXM service.  |
| NFXERR_INCOMPATIBLE_RECEIVER | 0x050216        | The application has disabled on-the-fly conversion and the receiver can not handle the specified image format.  |
| NFXERR_INTERNAL_ERROR        | 0x58000-0x5FFFF | An unexpected internal error has occurred.  |
| NFXERR_NEGOTIATION_FAILED    | 0x050202        | Failed T.30 fax negotiation at the beginning of a session or between documents. When transmitting, this error indicates that training failed (usually due to bad phone line quality) or that the transmitter and receiver capabilities do not match. When receiving, this error indicates that training failed. |
| NFXERR_NO_MODEMS             | 0x050213        | The required DSP files are not configured or loaded to the board. The FXM service failed to start.  |
| NFXERR_NO_PPM                | 0x050214        | The receiver has not received a correct post-page message.  |
| NFXERR_NO_PPM_RESPONSE       | 0x050215        | The receiver has not returned a correct response to the post-page message.  |
| NFXERR_NO_REMOTE             | 0x05021C        | The called fax terminal did not send a DIS signal.  |
| NFXERR_OPEN_QUEUE_FAILED     | 0x05020A        | Could not open the specified document queue. Check that you have a valid transmit queue.  |
| NFXERR_PROTOCOL_ERROR        | 0x050217        | A T.30 protocol error occurred.   |
| NFXERR_QUEUE_EMPTY           | 0x050206        | The specified document queue is empty.  |
| NFXERR_QUEUE_TOO_LATE        | 0x050208        | Attempted to enqueue a file in an active transmit queue too late in the fax session.  |

| Error or reason         | Hex      | Description  |
|-------------------------|----------|--|
| NFXERR_RATE_TOO_LOW     | 0x050210 | The receiving or transmitting fax terminal is attempting to receive or transmit at a rate lower than the minrate parameter.  |
| NFXERR_REMOTE_DCN       | 0x05021B | The remote terminal has sent an unexpected disconnect (DCN) command frame.   |
| NFXERR_RETRAIN_NEGATIVE | 0x05021E | The page was received with errors, and NaturalFax has requested that the remote transmitter retransmit the last page.  |
| NFXERR_SESSION_FAILED   | 0x050209 | The fax session failed due to an unexpected event. May indicate a T.30 timeout or failure to receive a post-page message. Check your host processor CPU and I/O loading. |
| NFXERR_T5_TIMEOUT       | 0x05021F | The receiver is not ready to receive after T5 (60 +/- 5 seconds).  |

## Informational events

The following events, presented in alphabetical order, provide additional information during a fax session:

| Event                   | Hex      | Description  |
|-------------------------|----------|--|
| NFXEVN_CANNOT_OPEN_FILE | 0x052007 | The specified file was not located by <a href="#">nfxSendFax</a> , <a href="#">nfxReceiveFax</a> , or <a href="#">nfxAnswerFaxPoll</a> . On transmit, the function continues to the next file in the document queue. |
| NFXEVN_DOC_BEGIN        | 0x052003 | Document transmission or reception started. After you receive this event, you can call <a href="#">nfxGetSessionStatus</a> to verify the negotiated values.  |
| NFXEVN_DOC_END          | 0x052004 | Document transmission or reception ended. To retrieve additional status information, call <a href="#">nfxGetSessionStatus</a> or <a href="#">nfxGetDocStatus</a> .   |



| Event                      | Hex      | Description   |
|----------------------------|----------|---|
| NFXEVN_PAGE_BEGIN          | 0x052001 | Page transmission or reception started. To retrieve additional status information, call <a href="#">nfxGetSessionStatus</a> . |
| NFXEVN_PAGE_END            | 0x052002 | Page transmission or reception ended. To retrieve additional status information, call <a href="#">nfxGetSessionStatus</a> .   |
| NFXEVN_POLLED              | 0x052005 | The calling fax machine polled NaturalFax.  |
| NFXEVN_PROCEDURE_INTERRUPT | 0x052006 | A procedure interrupt was received from the remote fax machine.   |
| NFXEVN_REMOTE_IDENTIFIED   | 0x05200A | The remote terminal information is available. You can check the session status to verify the remote SID and NSF.              |

### NFXEVN\_DOC\_END values

The following values can appear in the event value field of the NFXEVN\_DOC\_END event when NaturalFax successfully receives a fax:

| Value               | Hex    | Description  |
|---------------------|--------|--|
| SUCCESS             | 0      | Transmission of the document is complete.  |
| CTA_REASON_RELEASED | 0x1007 | The remote fax machine disconnected the call. Check that events are being handled in a timely fashion. Check your host processor CPU loading. The application will also receive ADIEVN_CALL_DISCONNECTED before NFXEVN_SESSION_DONE. |
| NFX_T30_EOM         | 1      | End of message received from remote fax terminal. More documents to follow.  |
| NFX_T30_EOP         | 2      | End of procedure received from remote fax terminal. No more documents will be transmitted.   |

### NFXEVN\_DOC\_END and NFXEVN\_PAGE\_END values

The following values can appear in the event value field of NFXEVN\_DOC\_END and NFXEVN\_PAGE\_END events when errors occur when NaturalFax is receiving a fax:

| Value                    | Hex               | Description   |
|--------------------------|-------------------|---|
| CTAERR_FILE_WRITE_FAILED | 0x28              | The file is not open or is locked, or the expected amount of data could not be written. Verify that an incorrect handle was not used to close another file.   |
| CTAERR_OUT_OF_MEMORY     | 0x6               | Unable to allocate memory for queue, driver or context, for play or record buffers, or for temporary storage. When this error occurs on a DONE event, it may mean that there was insufficient memory on the board.                          |
| CTAERR_OUT_OF_RESOURCES  | 0x8               | A NaturalFax DSP function could not be started. Check to see that you are using the proper board configuration for your application. Check that you have not exceeded the maximum number of fax operations for your hardware configuration. |
| CTA_REASON_RELEASED      | 0x1007            | The remote fax machine disconnected the call. Check that events are being handled in a timely fashion. Check your host processor CPU loading. The application will also receive ADIEVN_CALL_DISCONNECTED before NFXEVN_SESSION_DONE.        |
| CTA_REASON_STOPPED       | 0x1002            | The fax function was stopped by calling <a href="#">nfxStopSession</a> .  |
| NFXERR_CARRIER_LOST      | 0x05021D          | The received signal fell below the level specified by NFX_RECEIVE_PARMS.threshold, and the fax session was terminated.  |
| NFXERR_CONNECT_FAILED    | 0x050201          | The called fax machine was not detected. Check that the number you are dialing is connected to a fax machine.   |
| NFXERR_FAX_SERVICE       | 0x050219          | A fatal error occurred in the FAX service.  |
| NFXERR_FXM_SERVICE       | 0x05021A          | A fatal error occurred in the FXM service.  |
| NFXERR_INTERNAL_ERROR    | 0x58000 - 0x5FFFF | An unexpected internal error occurred.  |

| Value                     | Hex      | Description   |
|---------------------------|----------|---|
| NFXERR_NEGOTIATION_FAILED | 0x050202 | Failed T.30 fax negotiation at the beginning of a session or between documents. When transmitting, this error indicates that training failed (usually due to bad phone line quality) or that the transmitter and receiver capabilities do not match. When receiving, this error indicates that training failed. |
| NFXERR_NO_PPM             | 0x050214 | The receiver has not received a correct post-page message.  |
| NFXERR_PROTOCOL_ERROR     | 0x050217 | A T.30 protocol error occurred.   |
| NFXERR_QUEUE_EMPTY        | 0x050206 | The specified document queue is empty.  |
| NFXERR_RATE_TOO_LOW       | 0x050210 | The receiving or transmitting fax terminal is attempting to receive or transmit at a rate lower than the minrate parameter.   |
| NFXERR_REMOTE_DCN         | 0x05021B | The remote terminal has sent an unexpected disconnect (DCN) command frame.  |
| NFXERR_RETRAIN_NEGATIVE   | 0x05021E | The page was received with errors, and NaturalFax has requested that the remote transmitter retransmit the last page.   |
| NFXERR_SESSION_FAILED     | 0x050209 | The fax session failed due to an unexpected event. May indicate a T.30 timeout or failure to receive a post-page message. Check your host processor CPU and I/O loading.  |

## Confirmation events

The following events confirm the successful start of a NaturalFax session:

| Event                  | Hex      | Description  |
|------------------------|----------|--|
| NFXEVN_RECEIVE_STARTED | 0x052009 | Confirms that <b>nfxReceiveFax</b> or <b>nfxSendFax</b> (in response to a poll request) has started a receive operation. |
| NFXEVN_SEND_STARTED    | 0x052008 | Confirms that <b>nfxSendFax</b> or <b>nfxAnswerFaxPoll</b> has started.  |



## 12. Data structures

---

### Overview of NaturalFax data structures

NaturalFax stores system parameters, document status information, and fax session status information in its data structures. All NaturalFax data structures are defined in *nfxdef.h*. Refer to the files list for the NaturalFax's installed directory structure for the exact path to the NaturalFax header files.

NaturalFax uses data structures to specify the behavior of fax functions. Natural Access parameter functions retrieve and change the values of a specified parameter structure. If a function is governed by parameters, then one of its arguments is a pointer to the relevant parameter structure. You can fill in and pass a parameter structure in the function call, or accept the default parameter values by passing a NULL pointer.

The following Natural Access functions obtain or modify parameter information:

| Function                | Description  |
|-------------------------|--|
| <b>ctaGetParmByName</b> | Retrieves a single field for a given parameter name.                             |
| <b>ctaSetParmByName</b> | Modifies a single field for a given parameter name.                              |
| <b>ctaGetParmInfo</b>   | Retrieves a parameter field definition.  |
| <b>ctaGetParms</b>      | Returns parameter values for a given parameter structure.                        |
| <b>ctaRefreshParms</b>  | Resets the values of all context parameters on a context to the global defaults. |

For more information about parameter management in Natural Access, refer to the *Natural Access Developer's Reference Manual*.

## NFX\_TRANSMIT\_PARMS

The parameters in NFX\_TRANSMIT\_PARMS control the behavior of a fax transmission, including file encoding formats and resolution.

Dependent functions: nfxAnswerFaxPoll, nfxSendFax

| Field name    | Type  | Units            | Description   |
|---------------|-------|------------------|---|
| addheader     | DWORD | None             | <p>Mandatory for applications running in countries under FCC or DoC jurisdiction. Indicates whether headers are added to the transmitted image data. NFX_YES creates a header that is a fixed ASCII string in the following format:</p> <pre>FROM transmit_SID Date Time Page N of M</pre> <p>Values are:</p> <ul style="list-style-type: none"> <li>NFX_YES (default)</li> <li>NFX_NO</li> <li>NFX_CUSTOM</li> </ul> <p>For information on using NFX_CUSTOM, see custom_header.</p>  |
| custom_header | char  | ASCII characters | <p>Adds a customized header of up to 80 ASCII characters. The custom header can include page number and the date and time for each page using C style format strings:</p> <ul style="list-style-type: none"> <li>%d = Date/time string.</li> <li>%p = Page number.</li> <li>%P = Total number of pages in this fax.</li> <li>%y = Current page number in fax session.</li> <li>%Y = Total number of pages in all documents in queue.</li> </ul> <p>Default: None</p>  |
| encoding      | DWORD | None             | <p>Limits the image encoding format of fax images to be transmitted to 1D encoding. 1D encoding is supported by all Group 3 fax terminals. The actual encoding used depends on the value for this parameter, the encoding supported by the receiving fax terminal, and the encoding in the file to be transmitted. The useECM parameter must be set to NFX_YES to use NFX_ENCODE_MMR.</p> <p>Values are:</p> <ul style="list-style-type: none"> <li>NFX_ENCODE_1D (default)</li> <li>NFX_ENCODE_2D</li> <li>NFX_ENCODE_MMR</li> </ul> |

| Field name | Type  | Units         | Description   |
|------------|-------|---------------|---|
| ForceRate  | DWORD | None          | <p>If ForceRate is set to NFX_YES, then NFX negotiates the transmission protocol starting with the specified modemtype and minrate, rather than the default V.17 14400. During negotiation and training, NFX can drop to a lower modemtype and minrate, but not to a modem rate lower than the specified minimum rate.</p> <p>Default: NFX_NO</p>   |
| level      | INT32 | tenths of dBm | <p>The transmission level. Values are -150 to - 60.</p> <ul style="list-style-type: none"> <li>-150 corresponds to -15 dBm.</li> <li>-60 corresponds to -6 dBm.</li> </ul> <p>Default: -135</p>   |
| minrate    | DWORD | b/s           | <p>The minimum allowed rate that NaturalFax selects for fax transmissions. Can be used to limit the range of rates used.</p> <p>The transmit rate cannot drop below the minrate. Therefore, if the receiving fax terminal advertises only modems whose rates are below the minrate, the fax session immediately fails with an NFXEVN_SESSION_DONE event with NFXERR_RATE_TOO_LOW in the reason field.</p> <p>Values are:</p> <ul style="list-style-type: none"> <li>NFX_BIT_RATE_14400</li> <li>NFX_BIT_RATE_12000</li> <li>NFX_BIT_RATE_9600</li> <li>NFX_BIT_RATE_7200</li> <li>NFX_BIT_RATE_4800</li> <li>NFX_BIT_RATE_2400 (default)</li> </ul> |

| Field name | Type  | Units | Description  |
|------------|-------|-------|--|
| modemtype  | DWORD | None  | <p>The preferred modem for fax transmissions. Specifying a preferred modem sets an upper limit on the set of modems that NaturalFax considers when trying to match the receiving fax terminal's capabilities.</p> <p>Modems support the following rates of data transmission:</p> <ul style="list-style-type: none"> <li>• V.17 supports 7200, 9600, 12,000, and 14,400 b/s</li> <li>• V.29 supports 7200 and 9600 b/s</li> <li>• V.27ter supports 2400 and 4800 b/s</li> </ul> <p>If NaturalFax drops to a modem type that supports lower transmission rates, it attempts to use the highest rate for that modem type. If the transmitting fax terminal receives a retrain negative from the receiving fax terminal, and it is already using the rate specified by minrate, NaturalFax continues to use that rate. If not, NaturalFax attempts to use a lower transmission rate for the current modem. If the original rate is the lowest rate for the current modem, NaturalFax attempts to use the modem that supports the next lowest transmission rate that is still at or above the minrate value.</p> <p>Values are:</p> <ul style="list-style-type: none"> <li>• NFX_MODEM_TYPE_V27</li> <li>• NFX_MODEM_TYPE_V29</li> <li>• NFX_MODEM_TYPE_V17 (default)</li> </ul> |
| NSF        | BYTE  | bytes | <p>The value to be used in the non-standard facilities frame on transmit. Values are a byte array between 0 and 54 bytes in length.</p> <p>Default: NULL</p>   |
| NSFlength  | DWORD | bytes | <p>The length of the NSF field given in TRANSMIT_PARMS.NSF. If this parameter is non-zero, the value in NSF is used when negotiating with the remote fax terminal. Values are 0 - 96.</p> <p>Default: 0</p>  |



| Field name | Type  | Units | Description   |
|------------|-------|-------|---|
| OTFmode    | DWORD | None  | <p>Specifies when to perform on-the-fly conversion during a fax transmit operation. Values are:</p> <ul style="list-style-type: none"> <li>• NFX_OTF_NEVER: To disable on-the-fly conversion (default).</li> <li>• NFX_OTF_ONLY_IF_FAIL: To perform format conversions only when fax negotiations would otherwise fail.</li> <li>• NFX_OTF_ALWAYS: To always perform a conversion.</li> </ul> <p>For more information, refer to <a href="#">Image conversion during fax transmission</a>.</p> |
| pagewidth  | DWORD | None  | <p>The page width of fax images to be transmitted. The actual page width used depends on the value for this parameter, the page width supported by the receiving fax terminal, and the page width in the file to be transmitted. Values are:</p> <ul style="list-style-type: none"> <li>• NFX_PAGE_WIDTH_A4 (default)</li> <li>• NFX_PAGE_WIDTH_B4</li> <li>• NFX_PAGE_WIDTH_A3</li> </ul>  |
| PRIenabled | DWORD | None  | <p>Controls the ability of the transmitting fax terminal to send a procedure interrupt signal. When set to NFX_YES, a procedure interrupt (PRI-EOP) is sent to the called fax terminal after the last document in the transmit queue is transmitted with <a href="#">nfxSendFax</a>. This parameter has no effect when polling is used with <a href="#">nfxSendFax</a>.</p> <p>Default: NFX_NO</p>  |
| resolution | DWORD | None  | <p>Limits the maximum resolution of fax images to be transmitted. The actual resolution used depends on the value of this parameter, the resolution supported by the receiving fax terminal, and the image resolution in the file to be transmitted. Values are:</p> <ul style="list-style-type: none"> <li>• NFX_RESOLUTION_HIGH (default)</li> <li>• NFX_RESOLUTION_LOW</li> <li>• NFX_RESOLUTION_SUPER_HIGH</li> </ul>   |

| Field name    | Type  | Units   | Description  |
|---------------|-------|---------|--|
| retrainaction | DWORD | None    | <p>The action to be taken when receiver indicates that a page with too many bad lines was received. Values are:</p> <ul style="list-style-type: none"> <li>NFX_RTN_NEXT_PAGE: The next page in the TIFF-F file is transmitted (default).</li> <li>NFX_RTN_REPEAT_PAGE: The last page is retransmitted.</li> </ul>  |
| SID           | char  | None    | <p>The subscriber ID to be used when transmitting. The T.30 protocol recommends inserting the telephone number of the transmitting fax terminal in this field. In the US, the FCC requires using the telephone number in this field when transmitting faxes. Values are a character string of 0 to 20 characters in length. The T.30 protocol specifies a string of only digits 0 - 9, spaces, and [.] or [-].</p> <p>Default: NULL</p>  |
| SUB           | char  | None    | <p>The sub-address string to be used when transmitting. Values are a character string of 0 to 20 characters in length. The T.30 protocol specifies a string of only the digits 0 - 9, spaces, and * or #.</p> <p>Default: NULL</p>   |
| threshold     | N/A   | N/A     | Do not modify.   |
| timeout       | DWORD | Seconds | <p>The maximum allowed time for a transmitter to continue sending the CNG tone while waiting to receive a DIS tone from the receiving fax terminal. If a DIS tone is not detected before this time has expired, the NaturalFax transmitter cancels the fax operation. Values are 10 - 120.</p> <p>Changing this parameter makes the application non-compliant with phase B of the T.30 protocol, and greatly compromises fax machine compatibility. Use extreme caution when changing this parameter.</p> <p>Default: 35</p> |

| Field name | Type  | Units | Description   |
|------------|-------|-------|---|
| txrate     | DWORD | None  | <p>The preferred modem rate for the fax transmission. When the ForceRate flag is set to NFX_YES, NFX negotiates the transmission protocol starting with this modem rate. The parameter must be valid for the specified modem type. If an invalid txrate and modemtype combination is specified, the fax session terminates with a CTAERR_BAD_ARGUMENT error.</p> <p>If a modem type and rate are specified but the receiver is not compatible with the specified rate, NFX negotiates down to a lower modem rate. NFX does not negotiate to a modem rate lower than the specified minrate. Values are:</p> <ul style="list-style-type: none"> <li>• NFX_BIT_RATE_2400</li> <li>• NFX_BIT_RATE_4800</li> <li>• NFX_BIT_RATE_7200</li> <li>• NFX_BIT_RATE_9600</li> <li>• NFX_BIT_RATE_12000</li> <li>• NFX_BIT_RATE_14400 (default)</li> </ul> |
| useCNG     | DWORD | None  | <p>Controls whether a CNG tone is transmitted during call establishment (Phase A of the T.30 protocol). Values are:</p> <ul style="list-style-type: none"> <li>• NFX_YES (default)</li> <li>• NFX_NO</li> </ul>   |
| useECM     | DWORD | None  | <p>Controls whether the transmitter should use error correction mode (ECM) when it is available from the receiver. Values are:</p> <ul style="list-style-type: none"> <li>• NFX_NO: ECM is disabled (default).</li> <li>• NFX_YES: ECM is used if supported by the receiver.</li> </ul>   |
| useSUBADD  | DWORD | None  | <p>Controls whether a SUB is transmitted during negotiation (Phase B of the T.30 protocol). Values are:</p> <ul style="list-style-type: none"> <li>• NFX_YES</li> <li>• NFX_NO (default)</li> </ul>   |

## NFX\_RECEIVE\_PARMS

The NFX\_RECEIVE\_PARMS structure controls the behavior of fax reception, including the maximum data rate and the resolution accepted by the receiving fax terminal.

Dependent functions: nfxReceiveFax, nfxSendFax

| Field name    | Type  | Units         | Description  |
|---------------|-------|---------------|--|
| badlineaction | DWORD | None          | <p>The action that the receiving fax terminal takes upon receiving a bad line of image data. Values are:</p> <ul style="list-style-type: none"> <li>NFX_BAD_LINE_ACTION_NONE: Causes the bad line to be stored in the image file.</li> <li>NFX_BAD_LINE_ACTION_REPT: Repeats the previous line of image data (default).</li> <li>NFX_BAD_LINE_ACTION_DROP: Discards the line.</li> <li>NFX_BAD_LINE_ACTION_TICK: Adds a blank horizontal line with a tick mark in both margins.</li> </ul>   |
| encoding      | DWORD | None          | <p>The maximum image encoding format capabilities to be advertised when receiving. NFX_ENCODE_1D causes all files to be stored in a format supported by all Group 3 fax terminals. The useECM parameter must be set to NFX_YES to use NFX_ENCODE_MMR.</p> <p>NFX_ENCODE_TIFF_S causes the output file to be written using the TIFF-S specification. Values for the encoding, resolution, and pagewidth fields are overwritten with TIFF-S defaults of 1D encoding, low resolution, and A4 pagewidth.</p> <p>Default: NFX_ENCODE_1D</p> |
| level         | INT32 | tenths of dBm | <p>The transmission level. Values are -150 to -60. -150 corresponds to -15 dBm. -60 corresponds to -6 dBm.</p> <p>Default: -135</p>  |
| lineerrors    | DWORD | count         | <p>The threshold for bad lines, expressed as a percentage of lines received. If the percentage of bad lines is above this threshold, the receiver sends an RTN to request renegotiation with the transmitter. If this parameter is set to 100, NaturalFax never requests renegotiation. Values are 0 to 100.</p> <p>Default: 5</p>   |

| Field name | Type  | Units | Description   |
|------------|-------|-------|---|
| minrate    | DWORD | b/s   | <p>The minimum b/s rate to which the receiving fax terminal can negotiate. This parameter can be used to limit the range of rates used. Values are:</p> <ul style="list-style-type: none"> <li>NFX_BIT_RATE_14400</li> <li>NFX_BIT_RATE_12000</li> <li>NFX_BIT_RATE_9600</li> <li>NFX_BIT_RATE_7200</li> <li>NFX_BIT_RATE_4800</li> <li>NFX_BIT_RATE_2400 (default)</li> </ul>  |
| modemtype  | DWORD | None  | <p>The maximum modem capabilities that are advertised when receiving a fax. Specifying a preferred modem sets an upper limit on the set of modems that NaturalFax advertises. AG and CG boards are capable of V.17 receive.</p> <p>Modems support the following rates of data transmission:</p> <ul style="list-style-type: none"> <li>V.27 <i>ter</i> supports 2400 and 4800 b/s.</li> <li>V.29 supports 7200 and 9600 b/s.</li> <li>V.17 supports 14400, 12000, 9600, and 7200 b/s.</li> </ul> <p>Default: NFX_MODEM_TYPE_V29</p> |
| NSF        | BYTE  | bytes | <p>The value to be used in the non-standard facilities frame when receiving a fax. Values are a byte array between 0 and 54 bytes in length.</p> <p>Default: NULL</p>   |
| NSFlength  | DWORD | bytes | <p>The length of the NSF field given in RECEIVE_PARMNS.NSF. If this parameter is non-zero, the value in NSF is used when negotiating with the remote fax terminal. Values are 0 - 96.</p> <p>Default: 0</p>   |
| OTFmode    | DWORD | None  | <p>When to perform on-the-fly conversion during a fax receive operation. When receiving a fax, set to NFX_OTF_NEVER or NFX_OTF_ONLY_IF_FAIL to disable on-the-fly conversion and minimize CPU usage. Set to NFX_OTF_ALWAYS to always perform a conversion.</p> <p>For more information, refer to <a href="#">Image conversion during fax reception</a>.</p> <p>Default: NFX_OTF_NEVER</p>   |

| Field name     | Type  | Units         | Description   |
|----------------|-------|---------------|---|
| pagewidth      | DWORD | None          | Values are: <ul style="list-style-type: none"> <li>NFX_PAGE_WIDTH_A4 (default)</li> <li>NFX_PAGE_WIDTH_B4</li> <li>NFX_PAGE_WIDTH_A3</li> </ul>   |
| pollingenabled | DWORD | None          | Whether the receiver advertises the ability to respond to a poll request from the transmitting fax terminal.<br>Default: NFX_NO   |
| resolution     | DWORD | None          | The maximum resolution to be advertised by receiving fax terminal. Values are: <ul style="list-style-type: none"> <li>NFX_RESOLUTION_HIGH (default)</li> <li>NFX_RESOLUTION_LOW</li> <li>NFX_RESOLUTION_SUPER_HIGH</li> </ul>   |
| rewindonRTN    | DWORD | None          | Receive: discard page if retrain requested.<br>Default: NFX_NO  |
| SID            | char  | None          | The subscriber ID to be used when receiving. The T.30 protocol recommends using the telephone number of the transmitting fax terminal. In the US, the FCC requires using the telephone number in the SID field when transmitting faxes. Values are a character string of 0 to 20 characters in length. The T.30 protocol allows only the digits 0 - 9, spaces, and [.] or [-].<br>Default: NULL |
| SUB            | char  | None          | The sub-address string to be used when receiving. Values are a character string of 0 to 20 characters in length. The T.30 protocol allows only the digits 0 - 9, spaces, and * or #.<br>Default: NULL   |
| threshold      | INT32 | tenths of dBm | The lowest signal level to be accepted by NaturalFax as a receiver. Values are -300 to -430. The -430 value corresponds to a minimum signal level of -43 dBm.<br>Default: -430  |

| Field name | Type  | Units | Description  |
|------------|-------|-------|--|
| useECM     | DWORD | None  | Controls whether NaturalFax advertises error correction mode (ECM) capabilities as a receiver. If set to NFX_NO, ECM is not advertised.<br>Default: NFX_NO |
| useSUBADD  | DWORD | None  | Controls whether NaturalFax advertises SUB (sub-addressing) capabilities as a receiver. If set to NFX_NO, SUB is not advertised.<br>Default: NFX_YES       |

## NFX\_DOC\_PARMS

The NFX\_DOC\_PARMS structure specifies details about each document to be enqueued, including encoding format and page width.

Dependent function: nfxEnqueueDoc

| Field name | Type  | Units | Description  |
|------------|-------|-------|--|
| encoding   | DWORD | None  | The encoding format of specified document for the receiving fax terminal only. Encoding is used to specify how to store the received document when OTF conversions are enabled. This parameter has no effect during fax transmission, or when the OTF mode is set to NFX_OTF_NEVER. Values are: <ul style="list-style-type: none"> <li>• NFX_ENCODE_1D (default)</li> <li>• NFX_ENCODE_2D</li> <li>• NFX_ENCODE_MMR</li> </ul> |
| pagewidth  | DWORD | None  | The page width of the designated document. Page width specifies how to store the received document when OTF conversions are enabled. This parameter has no effect during transmit. Values are: <ul style="list-style-type: none"> <li>• NFX_PAGE_WIDTH_A4 (default)</li> <li>• NFX_PAGE_WIDTH_B4</li> <li>• NFX_PAGE_WIDTH_A3</li> </ul>   |
| resolution | DWORD | None  | The resolution of designated document for the receiving fax terminal only. The resolution parameter is used to specify how to store the received document when OTF conversions are enabled. This parameter has no effect during transmit. Values are: <ul style="list-style-type: none"> <li>• NFX_RESOLUTION_HIGH (default)</li> <li>• NFX_RESOLUTION_LOW</li> <li>• NFX_RESOLUTION_SUPER_HIGH</li> </ul>                     |

## NFX\_CONVERT\_PARMS

The NFX\_CONVERT\_PARMS structure specifies attributes that are used by **nfxConvertFileDirect**.

Dependent function: **nfxConvertFileDirect**

| Field name    | Type  | Units | Description   |
|---------------|-------|-------|---|
| badlineaction | DWORD | None  | <p>How bad lines that are discovered in the input file appear in the output file.</p> <ul style="list-style-type: none"> <li>NFX_BAD_LINE_ACTION_REPT: Repeats the picture elements of the nearest previous good line (default).</li> <li>NFX_BAD_LINE_ACTION_DROP: Eliminates the bad line from the destination file.</li> <li>NFX_BAD_LINE_ACTION_TICK: Replaces the bad line with a blank line with a short black mark at either end.</li> </ul> <p>When the source file has 2D encoding, an error in a single line may cause up to three subsequent lines to be considered bad.</p> |
| encoding      | DWORD | None  | <p>The encoding of the data to be stored in the output file. The input file data may be in 1D, 2D, or MMR encoding. NFX_ENCODE_TIFF_S writes the output file according to the TIFF-S specification. Values are overridden to 1D encoding, low resolution, and A4 page width.</p> <p>The encoding affects the size of image data only; it does not affect the appearance of the image. Values are:</p> <ul style="list-style-type: none"> <li>NFX_ENCODE_1D (default)</li> <li>NFX_ENCODE_2D</li> <li>NFX_ENCODE_MMR</li> <li>NFX_ENCODE_TIFF_S</li> </ul>                               |



| Field name                | Type                         | Units | Description  |                           |            |      |                              |     |                     |      |                           |
|---------------------------|------------------------------|-------|--|---------------------------|------------|------|------------------------------|-----|---------------------|------|---------------------------|
| pagewidth                 | DWORD                        | None  | <p>The page width of the data to be stored in the output file. The input file data may have any of the allowed page widths. The number of picture elements (pels) per line is 1728, 2048, and 2432 for NFX_PAGE_WIDTH_A4, NFX_PAGE_WIDTH_B4, and NFX_PAGE_WIDTH_A3, respectively. All are packed with the resolution of 200 picture elements per inch. Converting to a narrower page shrinks each line. Converting to a wider page results in padding the right side of the page with blank space. Values are:</p> <ul style="list-style-type: none"><li>NFX_PAGE_WIDTH_A4 (default)</li><li>NFX_PAGE_WIDTH_B4</li><li>NFX_PAGE_WIDTH_A3</li></ul> |                           |            |      |                              |     |                     |      |                           |
| resolution                | DWORD                        | None  | <p>The vertical resolution of the data to be stored in the output file. The input file data may have any of the allowed resolutions:</p> <table><tr><th>Scan lines per millimeter</th><th>Resolution</th></tr><tr><td>3.85</td><td>NFX_RESOLUTION_LOW (default)</td></tr><tr><td>7.7</td><td>NFX_RESOLUTION_HIGH</td></tr><tr><td>15.4</td><td>NFX_RESOLUTION_SUPER_HIGH</td></tr></table> <p>The horizontal resolution of the data is the same for all values. Image appearance is affected by a change in resolution.</p>  | Scan lines per millimeter | Resolution | 3.85 | NFX_RESOLUTION_LOW (default) | 7.7 | NFX_RESOLUTION_HIGH | 15.4 | NFX_RESOLUTION_SUPER_HIGH |
| Scan lines per millimeter | Resolution                   |       |  |                           |            |      |                              |     |                     |      |                           |
| 3.85                      | NFX_RESOLUTION_LOW (default) |       |  |                           |            |      |                              |     |                     |      |                           |
| 7.7                       | NFX_RESOLUTION_HIGH          |       |  |                           |            |      |                              |     |                     |      |                           |
| 15.4                      | NFX_RESOLUTION_SUPER_HIGH    |       |  |                           |            |      |                              |     |                     |      |                           |
| type                      | DWORD                        | None  | <p>The overall format of the output file. NaturalFax supports the TIFF-F format. Specifying a value of NFX_TIFF_S has the same effect as specifying a value of NFX_ENCODE_TIFF_S in the encoding field.</p> <p>Default: NFX_TIFF_F</p>   |                           |            |      |                              |     |                     |      |                           |

## Status structures

NaturalFax records the status of fax sessions and the status of each document processed during a fax session.

| Structure name   | Stores the status of...  |
|------------------|--|
| NFX_FAX_STATUS   | The currently active fax session.<br>The status information is current with respect to the most recent NaturalFax event. To view the structure definition, refer to the dependent function <a href="#">nfxGetSessionStatus</a> . |
| NFX_DOC_STATUS   | One document entry in a particular document queue.<br>To view the structure definition, refer to the dependent function <a href="#">nfxGetDocStatus</a> .  |
| NFX_CHECK_STATUS | The image format found on each page of a TIFF file.<br>To view the structure definition, refer to the dependent function <a href="#">nfxCheckTIFF</a> .  |

## 13. DSP requirements for AG and CG boards

---

### AG board DSP requirements for NaturalFax

One or more DSP processors on AG boards are devoted entirely to fax operations when the board is loaded. Each DSP processor configured for fax supports eight ports of any fax operation. For each DSP processor devoted to fax, add an additional line to the board keyword file. For example, to assign DSP 0 to fax, use this sample line in the board keyword file:

```
DSP.C5x[0].Image = ag2fax.c54  
DSP.C5x[0].Image = ag2faxes.c54
```

To determine if a board supports your application requirements, refer to the *NMS OAM System User's Manual* for IVR DSP requirements and the *Fusion Developer's Manual* for vocoder DSP requirements.

### CG board DSP requirements for NaturalFax

CG boards provide DSP processors that can simultaneously run multiple operations in a universal port model. Under this configuration, NaturalFax can support up to 120 ports of fax operation. These operations can include fax, a combination of other data processing functions (DPF), or both.

To run a fax operation, the NaturalFax DPF must be specified as one of the DSP files and as one of the resource definitions. For example, to load NaturalFax on DSPs 1 through 31, modify the board keyword as shown in the following example:

```
Resource[0].Definitions = (nmsfax | dtmf.det_all & callp.gnc & \  
                           (tone.gen | (rvoice.rec_mulaw & rvoice.play_mulaw) | (voice.rec_32 &  
voice.play_32_100)))  
Resource[0].DSPs = 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26  
27 28 29 30 31
```

## 14. Group 3 fax technology

---

### Fax protocols

The International Telecommunications Union (ITU) is the international standards body for telecommunications. In 1968 it published the standard for Group 1 (G1) fax protocol. In 1976 it published Group 2 (G2) protocol; in 1980 it first published Group 3 (G3) protocol.

Group 3 protocol is specified in several standards. T.4 and T.6 specify the image transfer protocols. T.30 specifies the session management procedures that support the establishment of a fax transmission. It allows the two stations to agree on such things such as transmission speed and page size. Since Group 3 protocol is specified for the switched analog network and is an all-digital procedure, it must use modems. The modems are specified in additional ITU standards:

- V.21 (300 b/s) for the T.30 procedures
- V.27<sub>ter</sub> (4800/2400 b/s) for image transfer
- V.29 (9600/7200 b/s) (optional)
- V.33 (14400/12000 b/s) (optional)
- V.17 (14400/12000/9600/7200 b/s) (optional)

The following International Telecommunications Union (ITU) specifications provide detailed information about facsimile:

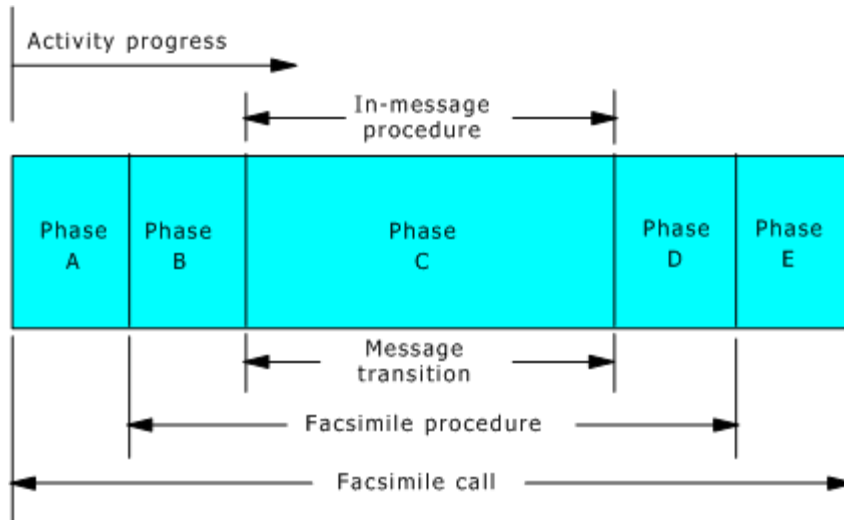
- ITU-T T.30: *Procedures for Document Facsimile Transmission in the General Switched Telephone Network*
- ITU-T T.4: *Standardization of Group 3 Facsimile Apparatus for Document Transmission*
- ITU-T T.6: *Facsimile Coding Schemes and Coding Control Functions for Group 4 Facsimile Apparatus*

### Group 3 protocol

To understand Group 3 fax protocol and the options it offers the system designer, it is important to understand the basics of T.30. T.30 sets the session-control procedures for Group 1, Group 2, and Group 3. It divides a call into five phases:

- Phase A - Call setup
- Phase B - Pre-message procedure
- Phase C - Image transfer and message transmission
- Phase D - Post-message procedures
- Phase E - Call release

The session-control procedures used to control the call from phase B to phase E (call release) use high-level data link control (HDLC) frames at 300 b/s as defined in V.21. The following illustration shows the call phases:



### Phase A - Call setup

Phase A verifies that a fax terminal is at each end of the transmission. Because Group 3 is intended for transmission over the voice network, the calling and called fax terminals send tones at the beginning of a fax call. The calling terminal periodically transmits a Calling tone (CNG - 1100 Hz for 0.5 seconds) that identifies it as a fax terminal. The called fax terminal answers with Called station identification (CED), a 2100 Hz tone that lasts for three seconds.

### Phase B - Pre-message procedure

Phase B identifies and selects the required facilities on the fax terminals involved in the transmission and negotiates which end will transmit, at what speed, for what page size, and so on. Phase B begins with the called station transmitting the mandatory facsimile control field, digital identification signal (DIS). This field is a packet that characterizes the called station's capabilities, including:

- Group
- Data rate
- Vertical resolution
- Image encoding
- Page width capabilities
- Maximum page length capability
- Handshake speed
- Error-correcting mode

DIS, and the other facsimile control fields (DTC and DCS), may each be associated with one or two optional packets. The first optional packet identifies the terminal sending the control field (station ID). The second optional packet is a non-standard facilities frame whose content is not specified by T.30. This non-standard field provides the opportunity for a fax terminal manufacturer to transmit customized data to remote fax terminals. You can use NaturalFax to set and read these fields in a standardized manner.

The calling fax terminal, which is still in control of the session, can request the called fax terminal to receive or to send. During normal fax operation, the calling fax terminal requests that the called fax terminal receive. During a fax polling operation, the calling fax terminal requests that the called fax terminal transmit.

If the calling fax terminal wishes to send, it determines the facilities of the called fax terminal from the received DIS, selects the parameters of the fax session, and sends the appropriate DCS frame to the called fax terminal. The called fax terminal acknowledges with a confirmation to receive (CFR).

To initiate a poll, the calling fax terminal sends a digital transmit command (DTC). As with DIS, the DTC can be sent with a station ID and the non-standard field. The non-standard field can be used to identify a particular fax document to be sent as well as the password of the polling fax terminal. If polled, the called fax terminal assumes control of the session. If it has nothing to send, it sends a disconnect to the calling fax terminal and hangs up. Therefore, the calling fax terminal should always send before polling. If the polled fax terminal has a document to send, it sends DCS, which is the command to receive.

## **Phase C - Image transfer and message transmission**

The Group 3 in-message procedures are specified in T.4 and T.6. The page is divided into horizontal picture elements (pels) of nominally 1728 pels/line of 215 mm, and vertical pels of either 3.85, 7.7, or 15.4 lines/mm (normal, high, and super high resolution). The minimum transmission time per line is specified in phase B.

T.4 specifies the G3 data-compression coding schemes, often referred to as Huffman encoding. One-dimensional (1D) run-length encoding involves fixed codes for white/black run lengths (for example, the number of contiguous white or black pels). Two-dimensional (2D) encoding provides additional compression by encoding two lines at a time; the second line specifies changes from the first. A special code is used for end of line (EOL). T.6 specifies the MMR coding scheme. In MMR, encoding lines are not separated by EOL codes. A sequence of six EOLs at the end of a page indicates the end of the page and marks the end of phase C.

Group 3 faxes can also be transmitted using the T.30 Annex A error-correcting standard. It uses the HDLC transmission control procedures used in the session-management procedures in phases B, D, and E to transfer the image data in phase C. HDLC includes cyclic-redundancy (CRC) error checking which allows the receiver to detect errors in the received image data. If there is an error in a received data packet, the transmitting end re-transmits at the request of the receiver (automatic repeat request - ARQ).

## **Phase D - Post-message procedures**

When document transmission is complete, the sending fax terminal can

- Return to phase B to switch directions (turnaround polling).
- Return to phase B to change resolution, paper size, or transfer data rate.
- Send an end of message (EOM) and return to phase B.
- Send a multi-page signal (MPS) and return to the beginning of phase C for the next page.
- Send an end of procedure (EOP) and proceed to phase E.

When document transmission is complete, the receiving fax terminal can

- Send a message confirmation (MCF) and proceed as instructed by the sending fax terminal's post-page message (EOM, MPS, or EOP).
- Request retraining, which forces a return to phase B.

### **Phase E - Call release**

The sending fax terminal sends an EOP frame, followed by a DCN (disconnect) frame. The sending and receiving fax terminals then release the call.

### **Non-standard facilities frame**

ITU T.30 allows a non-standard message (one not defined in T.300) to be included with most negotiations. These non-standard frames are called NSF when the called fax terminal sends DIS, NSS when the calling fax terminal responds to DIS with DCS, and NSC when commanding the called fax terminal to send (a poll).

A non-standard frame begins with the country code of the manufacturer, the manufacturer's code, and any additional ASCII information. Limited by a maximum allowable transmission time of three seconds per frame. The non-standard frame transmitted by the remote terminal is read only if the first three bytes match the country code and manufacturer's ID parameters for this terminal.

Some manufacturers use non-standard messages to negotiate use of proprietary features when two machines with the same manufacturer's ID code fax to each other.

## 15. Modem metrics

### Overview of modem metrics

Modem metrics provide information about the performance of the receive modems during a fax session. Use this information to determine the quality of current line conditions during a fax session.

NaturalFax modem metrics provide an estimate of the signal-to-noise ratio (SNR) and the results of the training check frame (TCF). These metrics are valid when NaturalFax is acting as the receive fax machine. Modem metrics are available to NaturalFax through the NFX\_FAX\_STATUS structure in the snr and rx\_training\_zeros fields.

```
typedef struct
{
    DWORD size;
    DWORD rate;
    DWORD ecm; /* Error correction mode: NFX_YES or NFX_NO */
    DWORD resolution;
    DWORD encoding;
    DWORD pagewidth;
    DWORD mode; /* NFX_MODE_IDLE, NFX_MODE_NEGOTIATING, */
                /* NFX_MODE_TRANSMITTING, NFX_MODE_RECEIVING, */
                /* NFX_MODE_DISCONNECTING, NFX_MODE_FINISHED */
    DWORD error; /* Last error code generated */
    DWORD docnumber; /* Current document in progress */
    DWORD pagenumber; /* Current page of the document */
    DWORD badlines; /* Number of bad lines (or bad frames in */
                  /* ECM mode) during current session */
    char filename[NFX_FILENAME_MAX];
                /* Name of current file processed */
    char remoteSID[NFX_MAX_SID];
                /* Received SID from remote station */
    BYTE remoteNSF[NFX_MAX_NSF];
                /* Received NSF from remote station */
    DWORD snr; /* Signal to noise ratio in dB */
    DWORD rx_training_zeros;
                /* Training zeros in the TCF */
                /* in tens of milliseconds */
    char remoteSUB[NFX_MAX_SUB];
                /* Received Sub-Address string from */
                /* remote station */
    DWORD sub_sent;
                /* Set if Sub-Address frame is sent to */
                /* remote station */
    DWORD modemtype; /* Modem used to transfer document */
} NFX_FAX_STATUS;
```

### Signal to noise ratio (SNR)

An estimate of SNR is calculated during two phases of a fax call, when NaturalFax is acting as the receive fax machine. These phases are the initial training check frame (TCF), and the subsequent page or image portion of the facsimile. At the end of each of these phases, a value of SNR is provided to give indication of the line quality during each phase.

The following table illustrates the valid ranges that NaturalFax reports for SNR, as well as what should be considered a good, marginal, or poor value for receive performance during a fax session:



| Line quality | SNR in dB |
|--------------|-----------|
| Poor         | 15 - 21   |
| Marginal     | 22 - 24   |
| Good         | 25 - 32   |

The values in the preceding table should be used as a guide when determining whether receive performance problems are being caused by line quality problems or compatibility issues. For example, if the fax session is training down to lower rates, failing, or aborting fax calls while receiving a fax, and the reported SNR is below 22 dB, the probable cause is poor line quality. However, if the same symptoms occur with an SNR above 25 dB, it is unlikely that line quality is the reason for failure.

## Training check frame (TCF)

The training check frame is the phase of a fax call when the sending fax transmits a sequence of zeros to the receiving fax machine at the highest common data rate negotiated during prior phases. This check determines whether the line quality is adequate to pass information at the desired rate. If the line quality is good, the receiving fax machine receives this sequence of zeros without error. If line quality is not good, bit errors will occur during reception, and the total amount of zeros is not received.

The duration of the zero sequence sent by the transmitting fax is 1.5 seconds. The T.30 protocol states that to successfully receive this training data, 1.5 seconds +/- 10% of these zero symbols should be received. If successful, the receiving fax sends back a confirm signal (CFR) to the transmitting fax. If the standard is not met, the receiving fax sends back a fail to train (FTT) signal, indicating to the transmitting fax that a lower data rate should be tried.

NaturalFax reports the results of training information. This information can be used to determine the quality of the training event, and therefore the quality of the line. This metric can be used to determine whether a receive problem is the result of poor line quality, insufficient training, or a compatibility issue. The rx\_training\_zeros metric reports the length of zeros received in tens of milliseconds. A good value for the TCF is greater than 1.35 seconds. If values less than this are reported, the receiver is not able to demodulate the incoming signal appropriately. This results in bit errors.

## Using SNR and TCF together

Using both SNR and TCF modem metrics, quick diagnostics of poor or failing receiver performance can be made. The following table shows how these two parameters can be used:

| Indication                             | SNR         | Training zeros      | Possible NFX receiver performance  |
|--|-------------|---------------------|--|
| Good line quality                      | >25 dB      | > 1.35 seconds      | No problems encountered.   |
| Good line quality<br>Non-compliant TCF | >25 dB      | 1.0 to 1.35 seconds | Possible FTT due to non-compliant sending fax.<br>Possible line errors in non-ECM mode, and retransmits in ECM mode. |
| Marginal line quality<br>Compliant TCF | 22 to 25 dB | 1.0 to 1.35 seconds | Possible FTT due to line conditions.<br>Possible line errors in non-ECM mode, and retransmits in ECM mode.           |
| Poor line quality<br>Compliant TCF     | <22 dB      | < 1.0 seconds       | Probable FTT due to poor line conditions.<br>Possible line errors in non-ECM mode, and retransmits in ECM mode.      |

## 16. Sub-addressing

---

### Overview of sub-addressing

The optional sub-addressing signal allows you to transmit or receive fax routing information during a fax session. The server can use the additional information to determine which extension should receive the fax. The SUB signal indicates that the following facsimile information field (FIF) information is sub-addressed in the called subscriber's domain. It can not be used to provide additional routing information in the facsimile procedure. SUB is only sent if bit 49 in the digital identification signal/digital transmit command (DIS/DTC) is sent.

The T.30 protocol recommends that the SUB signal contain 20 numeric digits. The least significant bit of the least significant digit is the first bit transmitted. Fill the unused octets in the information field with space characters, and right justify the information.

Sub-address information is available in the remoteSUB field of the NFX\_FAX\_STATUS structure. The sub-sent flag indicates whether or not the sub-address string was sent. This flag is set only when sub-address information is sent.

```
typedef struct
{
    DWORD size;
    DWORD rate;
    DWORD ecm; /* Error correction mode: NFX_YES or NFX_NO */
    DWORD resolution;
    DWORD encoding;
    DWORD pagewidth;
    DWORD mode; /* NFX_MODE_IDLE, NFX_MODE_NEGOTIATING, */
                /* NFX_MODE_TRANSMITTING, NFX_MODE_RECEIVING, */
                /* NFX_MODE_DISCONNECTING, NFX_MODE_FINISHED */
    DWORD error; /* Last error code generated */
    DWORD docnumber; /* Current document in progress */
    DWORD pagenumber; /* Current page of the document */
    DWORD badlines; /* Number of bad lines (or bad frames in */
                  /* ECM mode) during current session */
    char filename[NFX_FILENAME_MAX];
                /* Name of current file processed */
    char remoteSID[NFX_MAX_SID];
                /* Received SID from remote station */
    BYTE remoteNSF[NFX_MAX_NSF];
                /* Received NSF from remote station */
    DWORD snr; /* signal to noise ratio in dB */
    DWORD rx_training_zeros;
                /* Training zeros in the TCF */
                /* in tens of milliseconds */
    char remoteSUB[NFX_MAX_SUB];
                /* Received Sub-Address string from */
                /* remote station */
    DWORD sub_sent;
                /* Set if Sub-Address frame is sent to */
                /* remote station */
    DWORD modemtype; /* Modem used to transfer document */
} NFX_FAX_STATUS;
```

## Using sub-addressing

NaturalFax supports the sub-addressing feature for sending and receiving. By default, sub-addressing is enabled on the receive side and disabled on the sending side. To enable the sub-addressing feature, set useSUBADD to NFX\_YES in NFX\_TRANSMIT\_PARMS or NFX\_RECEIVE\_PARMS.

In NaturalFax demonstration programs, *nfxsend* uses *-a **subaddress*** to enable or disable the sub-addressing feature. If *-a **subaddress*** is used by *nfxsend*, a transmitter sends sub-address information to the receiver depending on the receiver's capability. If the receiver does not support this feature, the user is notified that sub-address information was not sent, but that the fax was sent.

The following example shows sub-addressing enabled:

```
nfxsend -plps0 -d555 -s0:0 -a 12549 -v sample.tif
nfxrecv -plps0 -s0:2 -v receive.tif
```

The following example shows sub-addressing disabled:

```
nfxsend -plps0 -d5555 -s0:0 -v sample.tif
```

## 17. Index

---

- 1**
- 1D encoding ..... 47
- 2**
- 2D encoding ..... 47
- A**
- A3 ..... 49
- A4 ..... 49
- ADI service..... 15, 27, 29
- adiAnswerCall ..... 29
- ADIMGR ..... 15
- adiStartCallProgress ..... 29
- adiStartToneDetector ..... 29
- AG boards ..... 17, 19, 124
- ASCII text file ..... 94
- B**
- B4 ..... 49
- board keyword files ..... 17, 20
- C**
- call control ..... 29
- call establishment ..... 29
- caller demonstration program ..... 84
- CG boards ..... 17, 19, 124
- CNG tone ..... 29
- completion events ..... 100
- configuration ..... 13, 14, 19, 20
- confirmation events ..... 107
- contexts ..... 27
- cover pages ..... 94
- cta.cfg ..... 14, 40
- CTA\_EVENT ..... 99
- ctaCloseServices ..... 43, 63
- ctaCreateContext ..... 27
- ctaCreateQueue ..... 27
- ctaInitialize ..... 26, 40
- ctaOpenServices ..... 27
- ctaSetTraceLevel ..... 40
- ctavers ..... 23
- ctaWaitEvent ..... 99
- ctdaemon ..... 14, 26, 40
- D**
- data structures ... 109, 110, 116, 120, 121
- DCN frame ..... 41
- demonstration programs .... 84, 86, 88, 90
- disconnect frame ..... 41
- document queue functions.. 62, 63, 65, 75
- document queues ..... 30, 42, 52
- DSP requirements ..... 124
- E**
- encoding format ..... 47
- environment variables ..... 13
- error handling ..... 41
- errors ..... 95, 97
- event queues ..... 27
- events ..... 99, 100, 104, 107
- F**
- fax and voice application ..... 25
- fax application ..... 10, 24
- fax protocols ..... 125
- FAX service ..... 15, 27
- fax session termination ..... 41
- faxback demonstration program ..... 86
- FAXMGR ..... 15
- functions ..... 54
  - document queues ..... 62, 63, 65, 75
  - fax and voice application ..... 25
  - fax application ..... 24
  - image formats ..... 58, 61
  - pages of a document ..... 69, 81
  - polling ..... 55
  - receive a fax ..... 71

|   |                    |                                     |         |
|---|--------------------|-------------------------------------|---------|
| send a fax .....                                      | 75                 | nfxCheckTIFF.....                   | 58      |
| status .....  | 66, 68             | nfxcnvrt utility .....              | 92      |
| stop a fax.....                                       | 82                 | nfxConvertFileDirect .....          | 61      |
| FXM service.....                                      | 15, 27             | nfxCreateQueue.....                 | 62      |
| <b>G</b>  |                    | nfxdef.h .....                      | 95, 109 |
| Group 3 protocol .....                                | 125                | nfxDestroyQueue .....               | 63      |
| <b>I</b>  |                    | nfxEnqueueDoc.....                  | 65      |
| image conversion .....                                | 33, 34             | NFXERR_XXXX .....                   | 95, 97  |
| image format.....                                     | 47, 50, 53, 58, 61 | NFXEVN_DOC_END .....                | 104     |
| informational events.....                             | 104                | NFXEVN_PAGE_END .....               | 104     |
| International Telecommunications Union<br>(ITU) ..... | 125                | NFXEVN_POLLED.....                  | 32      |
| ITU.....  | 125                | NFXEVN_SESSION_DONE.....            | 31, 100 |
| IVR.....  | 45                 | NFXEVN_XXXX.....                    | 104     |
| <b>M</b>  |                    | nfxGetDocStatus .....               | 66      |
| MMR encoding .....                                    | 47                 | nfxGetSessionStatus.....            | 68      |
| modem metrics.....                                    | 129                | NFXHEADERFONT .....                 | 13      |
| <b>N</b>  |                    | nfxmerge utility .....              | 93      |
| Natural Access configuration file.....                | 14                 | nfxMergeFile .....                  | 69      |
| Natural Access environment .....                      | 25                 | NFXMGR.....                         | 15      |
| Natural Access Server (ctdaemon).....                 | 14                 | nfxReceiveFax.....                  | 71      |
| NaturalFax .....                                      | 10, 13, 23         | nfxrecv demonstration program ..... | 88      |
| NFX service .....                                     | 15, 27             | nfxResetQueue .....                 | 75      |
| NFX_CHECK_STATUS.....                                 | 58, 123            | nfxsend demonstration program.....  | 90      |
| NFX_CONVERT_PARMS .....                               | 61, 121            | nfxSendFax .....                    | 75      |
| NFX_DOC_PARMS .....                                   | 34, 65, 120        | nfxsplit utility .....              | 93      |
| NFX_DOC_STATUS.....                                   | 34, 40, 66, 123    | nfxSplitFile .....                  | 81      |
| NFX_ENCODE_XXXX.....                                  | 47                 | nfxStopSession .....                | 82      |
| NFX_FAX_STATUS... 34, 40, 68, 123, 129,<br>132        |                    | nfxtxttf utility .....              | 94      |
| NFX_OTF_XXXX .....                                    | 34, 110            | NMSTEXTFONT.....                    | 13      |
| NFX_PAGE_WIDTH_XXX.....                               | 49                 | non-standard facilities frame ..... | 128     |
| NFX_RECEIVE_PARMS .....                               | 34, 71, 75, 116    | <b>O</b>                            |         |
| NFX_RESOLUTION_XXX .....                              | 49                 | OAM service .....                   | 15      |
| NFX_TRACE_XXX .....                                   | 40                 | OAMMGR.....                         | 15      |
| NFX_TRANSMIT_PARMS....                                | 34, 55, 75, 110    | offline image conversion .....      | 33      |
| nfxAnswerFaxPoll .....                                | 55                 | online image conversion.....        | 34      |
| nfxcheck utility .....                                | 92                 | on-the-fly image conversion .....   | 34      |
|   |                    | OTFmode parameter.....              | 35, 110 |

**P**

page width format ..... 49  
 pages of a document ..... 33, 53, 69, 81  
 parameter structures.. 110, 116, 120, 121  
 polling ..... 31  
 ports ..... 45  
 PRI (procedure interrupt request)..... 31

**Q**

QDIMGR..... 15

**R**

receive a fax..... 31, 52  
 receive queues..... 30  
 resolution format ..... 49

**S**

sample.tif..... 23  
 send a fax ..... 31, 52  
 send queues ..... 30  
 services ..... 15  
     closing ..... 43  
     opening..... 27  
 signal to noise ratio (SNR)..... 129, 131  
 SNR (signal to noise ratio)..... 129, 131  
 status information..... 53, 66, 68  
 status monitoring..... 40, 66, 68  
 status structures ..... 123  
 structures ..... 109

sub-addressing ..... 132, 133

**T**

T.30 ..... 125  
 T.37 ..... 51  
 TCF (training check frame) ..... 129, 130  
 terminating a fax session ..... 41  
 TIFF-F..... 47  
     functions ..... 58, 61, 69, 75, 81  
     image data ..... 50  
     utilities..... 84  
 TIFF-S ..... 51  
     functions ..... 61, 69, 81  
     generating files ..... 39  
     image data ..... 50  
     image format characteristics ..... 47  
     utilities..... 84  
 tracing..... 40  
 training check frame (TCF) ..... 129, 130

**U**

universal port ..... 45  
 UNIX environment variables ..... 13  
 utilities ..... 84

**V**

verifying NaturalFax ..... 23

**W**

Windows environment variables ..... 13