NmxToCD11

Version 1.12

User Guide

Nanometrics Inc.
Kanata, Ontario
Canada

NmxToCD11 Version 1.12 User Guide

The information in this document has been carefully reviewed and is believed to be reliable for Version 1.12.*xx*. Nanometrics, Inc. reserves the right to make changes at any time without notice to improve the reliability and function of the product.

Nanometrics, Inc.
250 Herzberg Road
Kanata, Ontario, Canada K2K 2A1
Tel (613)592-6776
Fax (613)592-5929
Email info@nanometrics.ca

Part number 14685R3
Release date 2005-04-22

Contents

Tables

NmxToCD11

1 About NmxToCD11

NmxToCD11 Version 1.12 forwards time series data from a Nanometrics data acquisition system to an IDC data center, using the CD-1.1 data format and protocol defined in IDC 3.4.3 Rev. 0.2 (December 2001) (http://www.rdss.info/). Each instance of the NmxToCD11 program represents a single IDC "station", which can include up to 100 data channels from different Nanometrics stations. Data are sent to the IDC as CD-1.1 data frames in near-real time; missing data are handled as described in Section 1.1.1.

This manual provides instructions for installing, configuring, and running NmxToCD11, and an overview of the NmxToCD11 log file and configuration file.

1.1 Typical operation

NmxToCD11 is designed to accept authenticated data from Nanometrics Europa and EuropaT digitisers. Each digitiser builds and signs CD-1.1 subframes in real time, then forwards the data and signed subframe headers separately to the NaqsServer data acquisition program in a proprietary Nanometrics data format. NaqsServer ensures that each data stream is complete, and maintains a ringbuffer (archive) containing the most recent several days of data for each channel.

NmxToCD11 subscribes to real-time data from NaqsServer via TCP socket using the Naqs Datastream interface, and maintains a second TCP connection to the IDC data center. NmxToCD11 assembles time series data and subframe headers for each channel into CD-1.1 frames. Frames have a fixed time duration (typically 10 seconds) and contain data for all channels comprising the IDC station. The data for each channel within a frame is called a channel subframe. During normal operation, CD-1.1 frames are built and sent to the IDC in near real time as online data are received from Naqs.

1.1.1 Handling outages

Occasionally data may be missing or delayed due to telemetry errors. In this case, NmxToCD11 will wait for missing data until a configurable timeout has expired, then send the incomplete frame. The incomplete frame will contain the channels with complete data; channels with missing samples are not sent (these channels will be sent later if the missing data are recovered).

If the connection to IDC is lost, or if the bandwidth of the connection is (temporarily) insufficient to handle the required data rate, data frames are buffered, then sent in reverse chronological order when the connection is reestablished.

For outage recovery, NmxToCD11 maintains a history file (`TxHistory.cd11`) that records the transmission status of each data frame. It keeps track of which channels were sent or not, and whether sent channels were complete or were missing frames.

NmxToCD11 also maintains a directory of frames that have been sent but not acknowledged, and a subdirectory of frames from previous connections that have never been acknowledged.

When the NmxToCD11 station is started, it updates the history files to include frames which should have been built while it was shut down and marks them as never sent. When NmxToCD11 has established a connection to the IDC receiver, it sends frames with the following priority within the available connection bandwidth:

1. Current data (real-time data), outage recovery data, and ReTx (gap response) data are all sent in reverse chronological order from a single time-ordered queue.

   Subframes for channels that were missing or incomplete when the frame was originally sent are rebuilt from data in the Naqs ringbuffers. If one or more of the missed channels are now complete, then a frame is sent containing only those channels which are now complete and were not sent previously.

2. Frames from the previous connection which were never acknowledged are sent only once, when the new connection is established.

1.2 Data compression

NmxToCD11 supports both of the data compression options specified in [IDC 3.4.3]. Data for each channel may be sent in uncompressed format (as 4-byte IEEE integers) or in Canadian compressed format (a second-difference compression format). The compression scheme is set via the NanometricsUI (Configuration > Authentication – Channel Description).

1.3 Authentication

Nanometrics Europa and EuropaT digitisers equipped with the authentication option provide data authentication following the IDC 3.4.3 format. Each channel subframe is signed with a 40-byte signature generated using the DSA signature algorithm. Data-signing capability within the digitiser is provided by an onboard PCMCIA encryption token. This provides complete security, because the private key used to sign the data cannot be exported from the token. NmxToCD11 also optionally generates a 40-byte DSA signature for each CD-1.1 frame with a token on its host workstation.

1.4 State of health

The CD-1.1 format as defined in [IDC 3.4.3] provides a status field within each channel subframe to define the channel status for the current frame. NmxToCD11 supports these defined channel status indicators:

- Clipped (bit 3 of data status byte) – This bit is set when the signal is clipped.

- Calibration underway (bit 4 of data status byte) – This bit can be mapped to an external SOH channel of the digitiser; it will be set if the scaled SOH value exceeds 1.0 at any time during the frame. The selected SOH channel should be used to monitor the calibration-enable signal on the digitiser, and scaled appropriately (for example, 0 = normal, 5 = calibration enabled). The Trident digitiser does not need a calibration SOH channel to be assigned.
- Vault door opened (bit 3 of channel security byte) – This bit can be mapped to an external SOH channel of the digitiser; it will be set if the scaled SOH value exceeds 1.0 at any time during the frame. The selected SOH channel should be used to monitor the vault door switch or transducer, and scaled appropriately (for example, 0 = closed, 5 = open).
- Authentication box has been opened (bit 4 of channel security byte) – This bit can be mapped to an external SOH channel of the digitiser; it will be set if the scaled SOH value exceeds 1.0 at any time during the frame. The selected SOH channel should be used to monitor the authentication box tamper switch or transducer, and scaled appropriately (for example, 0 = closed, 5 = open).
- Clock differential is too large (bit 1 of miscellaneous status byte) – This bit is set if the GPS is unlocked and the estimated maximum clock differential exceeds the configured limit.
- GPS receiver unlocked (bit 3 of miscellaneous status byte) – This bit is set if the GPS receiver is unlocked at any time during the frame.
- Supply voltage is out of range (bit 1 of voltage indicator byte) – This bit is set if the supply voltage to the digitiser is out of range at any time during the frame.
- Time of last GPS synchronization (offset 48 bytes from the beginning of the channel status field).
- Clock differential (in microseconds) at the end of the frame (offset 60 bytes from the beginning of the channel status field).

The assignment of some SOH channels to status bit and definition of some SOH thresholds are configured on the Europa digitiser (see Section 3.1 on page 5).

###temperature offset?

1.5 Automatic failover to backup system

On startup, each CPCSS CPU is automatically assigned either active or standby mode:

- The active CPU handles data transfer to the IDC.
- The standby CPU maintains a hot backup of the transmission state of the active side.

▸ The first time that the system is ever started, the active side should be started first.

The standby CPU switches to active after a configurable timeout of no activity from the active CPU (see also Table 3-1, “[Connections] section parameters,” on page 7).The operating mode (active or standby) is maintained in a file that is created on shutdown, therefore a failed-active CPU will assume standby mode automatically on restart by the absence of the mode file. The current operating mode of a CPU can be observed from its log output, where the standby CPU log contains only one type of operational message (“got state update” messages).

1.6 Summary of inputs and outputs

1.6.1 Required input files

- `NmxToCD11.ini` – This configuration file defines operating characteristics for the NmxToCD11 program.
- Naqs ringbuffers – NmxToCD11 requires read access to the Naqs ringbuffers for outage recovery. These files must either be stored on the same machine as NmxToCD11 or be accessible over a LAN on a shared drive.

1.6.2 Additional inputs

- Data from NaqsServer – NmxToCD11 receives online data from NaqsServer Datastream service. This requires a TCP connection to Naqs. See also the Nanometrics Data Formats reference guide for information on private data streams, and the NaqsServer manual.
- Encryption token – NmxToCD11 provides optional signing of CD11 frames; this authentication option requires an encryption token. See also the SMConsole manual.

1.6.3 Output files

- `NmxToCD11_`*yyyymmdd*`.log` – The log file contains diagnostic messages generated by NmxToCD11 and provides a summary of the program operation. Each log message has an associated type, ranked by severity (Table 1-1). Log verbosity can be configured to show only messages at or above a specified severity level.

  The verbosity of the log on startup is set in the [Log] section of the `NmxToCD11.ini` file (Section 3.4.3). While NmxToCD11 is running, you can set verbosity to a different level by using the run-time commands (Section 4.2).

Table 1-1 NmxToCD11 log message types

Label	Description
F	Fatal errors – Serious errors which cause immediate system shutdown.
E	Errors – Abnormal occurrences which will likely affect data integrity.
W	Warnings – Less serious abnormal occurrences.
I	Informational messages – Messages tracing the normal operation of the system.
V	Verbose messages – Detailed informational messages tracing the normal operation of the system.
D	Debug messages – Additional verbose trace messages.

Caution The output files `TxHistory.cd11`, `*.frm`, `SequenceNumbers.txt`, and `CalHistory.cd11` are used by the system. Do not edit these files.

- `TxHistory.cd11` (transmission history file) – The `TxHistory.cd11` file records the transmission status of each frame built by NmxToCD11. This information is used to determine which frames need to be retransmitted to IDC. The file is a ringbuffer which keeps the transmission status for the most recent few days; the duration of the file is set in the `NmxToCD11.ini` file (Section 3.4.2).

- To list the contents of the `TxHistory.cd11` file, run the program ViewHistory from the command line. See Section 4.4 on page 12; synopsis:
  `viewhistory` *filename* `[-b] [-s` *start time*`] [-d` *duration*`]`

- `*.frm` (frame files) – The unacknowledged frames storage directory contains `*.frm` files of frames that have been sent but not acknowledged. (Define this directory in the [Station] section of the configuration file.)
- `SequenceNumbers.txt` – NmxToCD11 creates the `SequenceNumbers.txt` file in the working directory and uses this file to keep track of frame sequence numbers.
- `calhistory.cd11` (calibration history file) – The `calhistory.cd11` file records the history of accepted calibrations broadcast from the Calibrate software, including updated *calib* (calibration factor) and *calper* (calibration period) values. You can view this file in any text editor.

2 Installing NmxToCD11

NmxToCD11 must be installed either on the NaqsServer computer, or on a computer that has TCP/IP access to the Naqs computer and network access to the Naqs ringbuffers.

- See the installation instructions for the acquisition system workstation.

3 Configuring your system for CD-1.1

To configure your data acquisition system to send authenticated data via CD-1.1, you must correctly configure the digitisers, NaqsServer, and NmxToCD11. This section provides the basic procedures for configuring each of these components. If you need more detailed information, see also the Nanometrics UI and NaqsServer user guides.

3.1 Configure the Europa/EuropaT digitisers for authentication

1. Start the Nanometrics UI and log on to the digitiser with `tech` access.
2. Click the Configuration tab to open the set of configuration panels.
3. Open the Authentication panel.
4. In the Signing section, enter values for:
   - the number of channels to sign
   - the CD1 frame duration
   - the login password for the token
   - the CD1 version: Choose CD1.1
5. In the Status Monitor section, enter values for:
   - voltage, temperature, and clock differential thresholds
   - the SOH channels to monitor calibration, vault door, and authentication box (Systems with Trident digitisers do not need a calibration SOH channel.)

   In each case, the selected SOH channel should be connected to the appropriate signal or transducer.

6. In the Channel Description section, enter values appropriate for your system into each of the fields.
7. Open the System panel. In the External SOH Calibration (units/volt) section:
   a) Enter values to set the SOH offset and calibration factors to scale the SOH values appropriately (see also the Nanometrics UI manual).

      For example, for a tamper switch, the switch closed condition should read 0 V and the switch open condition should read 5 V. The status bit will be set if the scaled SOH value exceeds 1.0.
   b) Edit each channel label as appropriate.
8. Open the Ringbuffers panel. Ensure that there is both a data ringbuffer and an authentication ringbuffer for each channel to be signed. For the authentication ringbuffers, each hour of data requires about 64 kilobytes.
9. Save the configuration:
   a) Click Submit.
   b) In the Submitting Config dialog box, click OK.
   c) When you are satisfied that the configuration is correct, click Commit to save to the digitiser flash memory.

3.2 Configure NaqsServer

NaqsServer must be configured to receive and archive both time-series data and subframe headers for each authenticated data channel. The subframe headers are treated as generic serial data by NaqsServer.

To configure NaqsServer to receive the subframe header channels, add the required entries to the `Naqs.stn` file:

1. Following each [ChannelPrototype] section, add one or more [SerialChannelPrototype] sections defining the data streams containing the subframe headers. In each prototype:
   a) Set *BytesPerPacket* to 153.
   b) Set the *Port* number to 16, 17, and 18 for data channels 1, 2, and 3 respectively (for example, authentication on data for sensor components Z, N, and E).
   c) Choose a unique *TypeName* to identify the resulting streams as subframe headers (for example, AUZ, AUN, and AUE for streams corresponding to the Z, N, and E components).
2. Following each [Instrument] section add one [SerialChannel] section for each authenticated data channel on this digitiser, to configure Naqs to receive the subframe headers.

3.3 Configure NmxToCD11

- Edit the configuration file `/nmx/user/NmxToCD11.ini` using values that are appropriate for your network.

  The configuration file is described in Section 3.4, and an example is given in Section A.2 on page 17.

3.4 Definition of configuration file sections and parameters

The `NmxToCD11.ini` file contains these sections:

- [Connections]
- [TxParameters]
- [Log]
- [Station]
- [Channels]

All sections are required. There must be exactly one section of each type. All sections, and all parameters within sections, must be in the order as listed below.

3.4.1 [Connections]

The [Connections] section defines the IP address and port for connecting to Naqs, the IDC Connection Manager, and the standby system. It contains the parameters described in Table 3-1.

Table 3-1 [Connections] section parameters

Parameter	Description
NaqsAddress	The IP address or host name of the NaqsServer machine. • Permitted values: a valid host name or dotted decimal IP address.
NaqsPort	The port number of the Naqs Datastream service. • Permitted values: a valid port number, typically 28000.
IdcAddress	The IP address or host name of the IDC Connection Manager. • Permitted values: a valid host name or dotted decimal IP address.
IdcPort	The port number of the IDC Connection Manager well-known port. • Permitted values: a valid port number.
CalMcastAddr	The multicast address on which to receive calibration updates. • Permitted values: a valid multicast address in dotted decimal format.
CalPortNum	The port number on which to receive calibration updates. • Permitted values: a valid port number.
MateAddress	The IP address of the remote active fault-tolerant mate accepting the connection. This is only needed on the connection initiating side. • Permitted values: a valid IP address in dotted decimal format or host name.
MatePort	The TCP port number connecting the two sides of the failover system. • Permitted values: a valid port number. Use `MatePort = 0` to run stand-alone.
ConnAcceptor	Indicates whether this computer is the connection acceptor or initiator. • Permitted values: 1 – accept connection, 0 – initiate connection. The parameter value must be different for each side.

Table 3-1 [Connections] section parameters (Continued)

Parameter	Description
FailoverTime	The amount of time to wait (in seconds) before declaring the remote side dead and doing a failover. • Permitted values: any positive integer, with the recommended minimum value of 60. ▸ Select a value that will allow brief temporary events (such as a user-initiated shutdown/restart of the active side) to proceed without causing a failover. ▸ To avoid a race condition in case of simultaneous startup of both main and standby systems, set this parameter to a value that is larger by a few seconds for the standby side than it is for the main side.
UpdateTime	How often (in seconds) to copy the state from the ACTIVE side to the STANDBY side. Select a value that accounts for the speed of the link between the sides. That is, for a slow link use a high enough value to allow an update to complete before starting the next update. • Permitted values: any positive integer.

3.4.2 [TxParameters]

The [TxParameters] section defines the CD-1.1 transmission characteristics of the station. It contains the parameters described in Table 3-2.

Table 3-2 [TxParameters] section parameters

Parameter	Definition
FrameTimeLength	The duration of each CD-1.1 data frame, in seconds. Each digitiser must be configured with the same *FrameTimeLength* to ensure transmission of properly authenticated data. • Permitted values: any integer from 5 to 60. Typical value is 10. • Recommended settings: 10 seconds for short-period or broadband seismic data, and hydroacoustic data; 20 seconds for long-period seismic data and infrasonic data.
MaxFrameDelay	The maximum time in seconds that a data frame will be delayed waiting for missing data. If this time is exceeded an incomplete data frame will be sent, containing only those channels for which all data have been received. Channels which are not sent will be sent later if the missing data are recovered. • Permitted values: any integer from 30 to 300. Typical value is 180.
TxHistoryHours	The duration in hours of the `TxHistory.cd11` file. Select a value that approximately matches the duration of the acquisition system data ringbuffer. • Permitted values: any positive float number.

Table 3-2 [TxParameters] section parameters (Continued)

Parameter	Definition

TxHistoryStart	An optional parameter that is used to define the start time of the `TxHistory.cd11` file. If *TxHistoryStart* is not used, the transmission history will be defined to start at the present time if the file does not already exist, and will remain unmodified if the file does exist.
	• *TxHistoryStart* can be used to define the period for which unsent data are recovered when NmxToCD11 is started. Typically, this is done in one of two ways:
	  • If `TxHistory.cd11` exists when NmxToCD11 is started, all frames with start time earlier than *TxHistoryStart* will be marked as sent and complete. The transmission history for frames later than *TxHistoryStart* will not be affected. This allows the system operator to limit the period for which catch-up frames are sent.
	  • If `TxHistory.cd11` does not exist when NmxToCD11 is started, then it will be created with a start time of *TxHistoryStart*, and all frames from *TxHistoryStart* to the present time will be marked as unsent. This allows the system operator to specify (on startup) a period for which catch-up frames should be sent.
	If *TxHistoryStart* is set to be later than the current time or earlier than the start time of an existing `TxHistory.cd11` file, it will be ignored. This ensures that the parameter takes effect only the first time that NmxToCD11 is started. Restarting NmxToCD11 with the same value of *TxHistoryStart* will not cause any further modification of the history file.
	• Permitted values: a valid date in format *yyyy-MM-dd-HH-mm-ss*, or the parameter is not used.
	Example:
	`TxHistoryStart = 2004-04-21-12-00-00 //standard format`
	or
	`TxHistoryStart = 2004/04/21 12:00:00 //alternate format`
	or comment out the line to disable this parameter:
	`//TxHistoryStart = 2004-04-21-12-00-00 //not used`

3.4.3 [Log]

The [Log] section defines the name and location of the NmxToCD11 log file and its verbosity setting at startup. It contains the parameters described in Table 3-3. (See also Section A.1, “Log file overview”.)

Table 3-3 [Log] section parameters

Parameter	Description
LogFilename	The base name for the NmxToCD11 log file. NmxToCD11 creates a new log file every day. The log file name is determined by inserting the date in format *yyyymmdd* between the base name and the file extension; for example, `NmxToCD11_20040419.log`. • Permitted values: any valid file name, with no spaces. Example: `LogFilename = NmxToCD11.log`
LogDirectory	The directory in which to store the NmxToCD11 log file. • Permitted values: any valid directory name, with no spaces. Do not use a trailing slash.

Table 3-3 [Log] section parameters (Continued)

Parameter	Description
Verbosity	The startup verbosity of the NmxToCD11 log file. • Permitted values: FATAL, ERROR, WARNING, INFO, VERBOSE, or DEBUG (see also Table 1-1 on page 4). Typical value is INFO.

3.4.4 [Station]

The [Station] section defines parameters such as state-of-health thresholds which apply to the entire IDC station; that is, to all channels. It contains the parameters described in Table 3-4.

Table 3-4 [Station] section parameters*

Parameter	Definition
TokenPIN	The password required to log on to the token.
UnackedStorageDir	The directory in which to store frames that have been sent but which have not yet been acknowledged. • Permitted values: a valid directory name, with no spaces. Example: `UnackedStorageDir = ./Frames`
NaqsRunDir	The absolute pathname to the Naqs running directory. • Permitted values: a valid pathname, with no spaces. Do not use a trailing slash. Example: `NaqsRunDir = /nmx/user`

* The parameter *AuthKeyID* has been removed starting with Version 1.12. NmxToCD11 will ignore it if it is included in the configuration file.

3.4.5 [Channels]

Note Do not list the authentication channels, as they are derived automatically.

The [Channels] section defines the list of Naqs data channels to build into CD1.1 data frames. Each channel must be listed on a separate line, and must be entered as a string in dotted format. For example:

```
Eu101.BHZ
Eu101.BHE
Eu101.BHN
```

The channel name must be defined in the NaqsServer `Naqs.stn` file.

4 Running NmxToCD11

In a typical network, NmxToCD11 will be set up to start automatically using scripts (on Solaris and Linux), or the NmxWatchdog program (on Windows). It can also be started manually from the command line. Once NmxToCD11 is running, you can use the console window options to change the log message verbosity, to create a new log file, to restart NmxToCD11, and to stop NmxToCD11.

4.1 Starting and stopping NmxToCD11 manually

Solaris and Linux:

- To start NmxToCD11, enter `nmxtocd11 start` in any terminal window.
- To stop NmxToCD11, enter `nmxtocd11 stop` in any terminal window.

Windows:

- To start NmxToCD11, enter `nmxtocd11` in any command window.
- To stop NmxToCD11, enter `quit` in the NmxToCD11 console window.

4.2 Using the NmxToCD11 run-time commands

The NmxToCD11 terminal window displays log messages generated by the NmxToCD11 program (see Section A.1, “Log file overview” for a description of log messages). It also supports a basic keyboard interface, with the options described in Table 4-1.

- On Solaris and Linux, you can run these commands from any terminal window. Enter `nmxtocd11` *command*
- On Windows, run these commands from the NmxToCD11 command window. Enter *command*

Table 4-1 NmxToCD11 command options

To do this...	Enter this command...
Generate all log messages to the log file; set the log verbosity to DEBUG	`D`
Suppress debug messages in the log file; set the log verbosity to VERBOSE	`V`
Suppress debug and verbose messages in the log file; set the log verbosity to INFO	`I`
Move the log file (close the current log and start a new file)	`M`
Restart NmxToCD11	`restart`
Stop NmxToCD11 and exit	`Quit`

4.3 Managing NmxToCD11 remotely

You can enable the CD11 sender via the AutoDRM `START_CONTINUOUS` command, and disable the CD11 sender via the AutoDRM `STOP_CONTINUOUS` command.

If running in fault-tolerant mode with 2 CPUs, the standby side must be shut down first. This requires AutoDRM to be running on both CPUs.

4.4 Viewing the TxHistory file

To list the contents of the transmit history file, run the program ViewHistory from the command line. (In Windows, run ViewHistory in the working directory. In Solaris and Linux, run ViewHistory from any terminal window):

- To list the viewhistory options, enter `viewhistory`
- To view a brief summary of the TxHistory file, enter `viewhistory` *filename* `-b`
- To list all entries in the TxHistory file, enter `viewhistory` *filename*
- To list a segment of the TxHistory file starting from a specific time, enter `viewhistory` *filename* `-s` *yyyy-mm-dd-hh-mm-ss*
- To list a segment of the TxHistory file for a particular period (in seconds) back from the last entry, enter `viewhistory` *filename* `-d` *duration in seconds*

4.5 Monitoring NmxToCD11 operation

4.5.1 Normal operation

Log messages generated by NmxToCD11 provide a summary of the program operation. If the log verbosity is set to INFO, general informational messages and all warning, error, and fatal error messages will be logged. (See also Section 3.4.3, "[Log]," on page 9 and Section A.1, "Log file overview," on page 15.)

If everything is OK, you will see the following startup messages:

```
MessageClient...(2) Connected to Naqs (address:port)
ConnectionMgr...(2) Connected to WellKnownPort (address:port)
ConnectionManage(2) Connected to Receiver (address:port)
```

and then NmxToCD11 will be fairly quiet with only frame Rx/Tx activity printed at the console window. NmxToCD11 also prints an hourly report of transmission results as an INFO message. Change the log verbosity to VERBOSE to display more details. An excerpt of a log is shown below (with *FrameTimeLength* of 10 seconds):

```
V 2003-02-06 22:55:06  FrameBuilder....(1) buildFrame: 2003-02-06
   22:54:50.000 Chans:3 FrmSize:3964
V 2003-02-06 22:55:12  RTBuildMgr......(1) New builder (3): 3/3
   2003-02-06 22:55:10.000 latency 2
V 2003-02-06 22:55:13  FrameBuilder....(1) buildFrame: 2003-02-06
   22:55:00.000 Chans:3 FrmSize:3964
I 2003-02-06 22:55:21  ConnectionMgr...(1) Rx AckNackFrm  Seq#:0
   FrmSet:WRA:IDC Lo#:0 Hi#:-1 Gaps:0
I 2003-02-06 22:55:21  ConnectionMgr...(1) Rx AckNackFrm  Seq#:0
   FrmSet:WRA:0 Lo#:919 Hi#:933 Gaps:0
I 2003-02-06 22:55:28  ConnectionMgr...(1) Tx DataFrame   Seq#:935
   2003-02-06 22:55:10.000 Chans:3 FrmSize:3964
V 2003-02-06 22:55:34  RTBuildMgr......(1) New builder (3): 3/3
   2003-02-06 22:55:30.000 latency 4
```

These messages show the frame type, the number of complete channels, and the nominal frame start time (*YYYY-MM-DD*, *HH:mm:ss*). For real-time frames, a channel count less than the expected total number of channels indicates that data are being dropped in the telemetry system (these data will be sent later in catch-up frames).

4.5.1.1 Automatic failover

When running 2 CPUs in automatic failover mode, in addition to the normal operational messages shown above, the active side will also display messages about the state of the mate connection and the remote mate. Once the mate connection is established, both sides will display a message once per update period (configured in the `.ini` file [Connections] section) indicating the successful transfer of the application state information from the active to the standby side.

4.5.2 Troubleshooting

4.5.2.1 Configuration file error detection

NmxToCD11 parses the `.ini` files on startup. If it detects any errors—for example, unrecognized fields or illegal values—it will print an error message to the log and then stop. To resume, fix the `.ini` file using a text editor (Section A.2 on page 17), then restart NmxToCD11.

- The most common causes of unrecognized fields are:
  - Misspelled parameter names – Check the spelling carefully, and note that parameter names are case-sensitive.
  - Missing names – If a parameter appears out of order or in the wrong section, it will not be recognized.
  - Duplicated names – If a parameter name appears more than once (or more than once within a section if multiple sections are permitted), instances of the parameter after the first instance will not be recognized.
- Illegal values are values which are undefined or out of range for a particular parameter. The permitted values for each parameter are given in Section 3.4.

4.5.2.2 Check the connection to NaqsServer

When NmxToCD11 connects to NaqsServer, you will see a log message like this:
`MessageClient...(2) Connected to Naqs (`*address:port*`)`

where *address:port* is the IP address or host name and port of the NaqsServer Datastream service.

If NmxToCD11 cannot connect to NaqsServer, this error message will be logged:
`Failed to connect to NaqsServer %reason for failure%`

- If NmxToCD11 cannot connect to NaqsServer, ensure that NaqsServer is running and that the Naqs address and port are specified correctly in the `NmxToCD11.ini` file (Table 3-2, “[TxParameters] section parameters,” on page 8).

4.5.2.3 Check the connection to IDC

When NmxToCD11 connects to the IDC Connection Manager, you will see a log message like this:
`ConnectionManage(1) Connected to WellKnownPort (`*address:port*`)`

where *address:port* is the IP address and port of the IDC connection manager.

When NmxToCD11 connects to the IDC CD1.1 Receiver, you will see a log message like this:
`ConnectionManage(2) Connected to Receiver (`*address:port*`)`

where *address:port* is the IP address and port of the IDC CD1.1 Receiver.

If NmxToCD11 cannot connect to IDC, one or more of the following messages will be reported:
```
Exception in connectToWellKnownPort()
Exception in connectToReceiver()
```

- If NmxToCD11 cannot connect to the IDC, check that these conditions are met:
  - The IP address and port of the IDC are specified correctly in `NmxToCD11.ini`.
  - A network connection exists between the NmxToCD11 machine and IDC.
  - TCP connections from NmxToCD11 to IDC are enabled by all intervening firewalls.
  - The IDC Connection Manager and CD1.1 Receiver are both running.

4.5.2.4 Check that data are getting to Naqs

If data or subframe headers for one or more channels are not arriving in real time, frames built by NmxTOCD11 will be delayed by *MaxFrameDelay* (Table 3-2, "[TxParameters] section parameters," on page 8) and will contain fewer than the set of channels configured. This probably indicates a telemetry problem between the specified station and Naqs. You can use the Waveform program to check Naqs operation.

Appendix A Log File and Configuration File

This section provides an overview of the log file, including a description of some of the log messages, and of the NmxToCD11 configuration file, including editing conventions and an example.

A.1 Log file overview

The log file contains diagnostic messages generated by NmxToCD11 and provides a summary of the program operation. Each log message has an associated severity or verbosity level (Table 1-1, "NmxToCD11 log message types," on page 4).You can configure the log to show only messages at or above a specified severity level by adjusting the verbosity setting (Section 3.4.3 and Section 4.2).

- Informational messages include info, verbose, and debug messages. These report the normal operation of the program. For example, an info-level message is written to the log every time that a frame is built or sent.
- Error messages severity levels include fatal, error, and warning. See Table A-1 for descriptions of a subset of the log error messages.

A.1.1 Error messages

Table A-1 Subset of error messages

Log message	Type	Description
Bad Crc	Error	The crc of a received frame was not correct.
Error signing frame: Bad signature length	Error	There was an error creating a signature for a frame being sent. ▸ If the problem persists, check the token.
Cannot open RBF File	Error	Cannot open the specified ringbuffer file; NmxToCD11 will be unable to reconstruct data for the affected channels. ▸ If this error occurs, check the configuration ([Station] section parameters). If the error persists, it may indicate a problem such as a disk error.

Table A-1 Subset of error messages (Continued)

Log message	Type	Description
ConnectionMgr is not Running Ok	Error	ConnectionMgr thread died; NmxToCD11 will shut down. ▸ Restart NmxToCD11 (see Section 4 on page 10)
Couldn't put frame back in UnackedPrevConn	Warning	After a failed resend, could not put the frame back in the unacknowledged frame storage area. The frame will be reconstructed later from the ringbuffer.
Data channel <channel> not found in Naqs channel list	Error	The specified channel name is not available from Naqs. ▸ Correct the channel name in the `NmxToCD11.ini` file.
Error initializing Seq Num File	Error	Problem creating the sequence numbers file; this can occur on startup due to external problems such as low memory or a full disk.
Error in existing history file	Warning	Indicates that the history file is corrupted and will be rebuilt. All transmission history information will be lost.
Error in sending frame	Warning	Indicates a write error on the TCP connection to IDC. The connection will be closed, then reopened again after 60 seconds.
Error moving log file	Error	The program is unable to move the log file, on receiving a move command.
Error Reading Sequence Number File	Error	Could not read sequence numbers file. The file may be corrupted.
Error receiving frame	Warning	Indicates an error receiving a frame on the TCP connection from the IDC.
Failed to connect to NaqsServer	Error	Failed to connect to Naqs Server. ▸ Ping the Naqs server machine; check Naqs parameters in [Connections] section in `NmxToCD11.ini`
FrameScheduler is not Running Ok	Error	FrameScheduler thread died; NmxToCD11 will shut down. ▸ Restart NmxToCD11 (see Section 4 on page 10).
Heartbeat Timeout: closing connection	Error	Have not received an AK frame in the required time so will close connection.
Invalid console input	Warning	Indicates an unrecognized command entered at the console window. The command will be ignored.
Invalid FrameSet - using default frameset	Warning	Encountered an invalid frameset when constructing an AK frame; using default frameset.
Invalid FrameSet received: ignoring	Warning	Received invalid frameset; ignored frameset.
MAX_BUILDERS limit reached	Warning	Cannot build this frame as builders are all busy; will rebuild later from ringbuffer.
MessageClient is not Running Ok	Error	MessageClient thread died; program will shut down. ▸ Restart NmxToCD11 (see Section 4 on page 10).
NmxToCD11 configuration error	Fatal error	The configuration file contains errors. ▸ Edit the `NmxToCD11.ini` file, then retry.
RbfFrameBuilder is not Running Ok	Error	RbfFrameBuilder thread died; NmxToCD11 will shut down. ▸ Restart NmxToCD11 (see Section 4 on page 10).

Table A-1 Subset of error messages (Continued)

Log message	Type	Description
RTBuildMgr is not Running Ok	Error	RTBuildMgr thread died; NmxToCD11 will shut down. ▸ Restart NmxToCD11 (see Section 4 on page 10).
Rx Unknown Frame	Warning	Received a frame that is not a CD11 defined frame type; frame is ignored.
Sequence Number File Not Found	Error	Sequence number file cannot be located. It may have been deleted or moved.
Serial channel <channel> not found in Naqs channel list	Error	The specified channel name is not available from Naqs. ▸ Correct the channel name in the `NmxToCD11.ini` file.
Thread died ... quitting ...	Fatal error	A component thread has stopped running properly. ▸ Restart NmxToCD11 (see Section 4 on page 10).
Unable to open history file	Error	Cannot open the history file. The program will exit. ▸ Move or delete the history file, then retry.

A.2 Configuration file overview

The main purposes of the configuration file `NmxToCD11.ini` are to provide connection information (addresses and ports) for IDC and Naqs, and detailed CD-1.1 transmission parameters

A.2.1 Editing the NmxToCD11 configuration file

`NmxToCD11.ini` uses a text file format which is designed to be readable and easy to edit. It consists of a number of sections, each containing several parameters. Sections are identified by a name enclosed in square brackets; for example, [Connections]. Each parameter is listed on a separate line following the section identifier, typically in the format *Parameter = Value*.

For example, part of a section defining network connections for NmxToCD11:

```
[ Connections ]
NaqsAddress = 199.71.138.13
NaqsPort = 28000
IdcAddress = 193.81.205.6
IdcPort = 9050
```

A.2.1.1 Data order and default values

All parameters for a given section must appear after the section identifier for that section, and before any other section identifier. All sections, and all parameters within sections, must be in the order as listed in Section 3.4, “Definition of configuration file sections and parameters,” on page 7.

A.2.1.2 White space and comments

The inifile reader ignores white space and blank lines, so white space can be added anywhere within the inifile if desired to improve readability. Also, the inifile reader recognizes the double-slash // as a comment delimiter, so comments can be added anywhere

in the file. Comments are useful for adding descriptive information to the file. For example

```
// This is a full line comment.

[ Interface ] // a comment can follow a section header
NaqsAddress = 199.71.138.13 // a comment can follow a parameter definition
```

Use comment delimiters for temporarily removing parameters or sections from the file. For example:

```
//TxHistoryStart = 2003/02/07 12:00:00 // this parameter is inactive
```

A.2.2 Example `NmxToCD11.ini` file

```
//=============================================================================
// Configuration file for NmxToCD11 data conversion program
//=============================================================================

[ Connections ]

NaqsAddress = 199.71.138.168 // Naqs server IP address or host name
NaqsPort    = 28000          // Naqs server port number

IdcAddress = 199.71.138.168  // IP address or host name of IDC Connection Manager
                             // Well Known Port
IdcPort    = 2000            // port number of IDC Connection Manager Well Known Port

CalMcastAddr = 230.1.2.1 // Multicast address on which to receive Calibration updates
CalPortNum   = 32000     // port number on which to receive calibration updates
MateAddress  = localhost // IP address of the remote ACTIVE fault-tolerant mate
MatePort     = 00000     // TCP port number connecting the two sides
                         // (0 = run stand-alone)
ConnAcceptor = 1         // 1 = accept connection, 0 = initiate connection (must be
                         // different for each side)
FailoverTime = 60        // seconds to wait before declaring the remote side dead
                         // and doing a failover. This value should differ for the two
                         // sides by a few seconds.
UpdateTime   = 60        // how often to copy the state from the ACTIVE to the
                         // STANDBY side (in seconds)

[ TxParameters ]

FrameTimeLength = 10                   // in seconds
MaxFrameDelay  = 180                   // in seconds (not less than 10 seconds)
TxHistoryHours = 168                   // duration of the tx history file in hours
//TxHistoryStart = 2003-02-21 20:40:00 // (optional) start time of TxHistory

[ Log ]

LogFilename = NmxToCD11.log
LogDirectory = /nmx/user/logs
Verbosity = VERBOSE

[ Station ]

TokenPIN  = nmxnmx             // the password required to access the token
UnackedStorageDir = ./Unacked  // dir to store frames that haven't been acked
NaqsRunDir        = /nmx/user  // absolute path to the directory that naqs is
                               // running in

[ Channels ]  // list of the naqs channels to build into data frames

Eu304.BHZ     // Nanometrics dotted station-channel name
```

```
Eu304.BHE
Eu304.BHN

Ja001.BHZ
Ja001.BHE
Ja001.BHN
```