Qualcomm Technologies, Inc.

QXDM Professional™ v5 for Windows OS

User Guide

80-V1241-25 E

August 12, 2021

Qualcomm Technologies, Inc.
5775 Morehouse Drive
San Diego, CA 92121
U.S.A.

Revision history

Revision	Date	Description
A	September 2015	Initial release
B	March 2016	Added the manage configuration option in Section 3.1; added Sections 3.3.3 and 3.3.5; updated the graph view image
C	January 2017	Numerous changes have been made to this document; it should be read it its entirety.
D	TBD	Added information to activate QXDM using a WIBU key
E	April 2019	Modify it for QXDM5
F	November 2019	Removed most of Automation Interface API (moving to QUTS)
G	August 2021	QXDM5 no longer supporting windows OS older than Windows 10

Contents

Figures

Tables

1 Introduction

1.1 Purpose

The QXDM Professional™ (QXDM Pro) tool provides a diagnostic client for dual-mode subscriber station (DMSS), newer user equipment (UE) software, and advanced mobile subscriber station (AMSS)

QXDM Pro provides a rapid prototyping platform for new diagnostic clients and diagnostic protocol packets. It provides a graphical user interface (GUI) that displays data transmitted to and from the DMSS.

NOTE: The use of DMSS and AMSS in this document refers loosely to user equipment that is connected to QXDM Pro using the Qualcomm Technologies, Inc. (QTI) diagnostic interface.

1.2 Conventions

Function declarations, function names, type declarations, attributes, and code samples appear in a different font, for example, `#include`.

Code variables appear in angle brackets, for example, `<number>`.

Commands to be entered appear in a different font, for example, **`copy a:*.* b:`**.

Button and key names appear in bold font, for example, click **Save** or press **Enter**.

Shading indicates content that has been added or changed in this revision of the document.

1.3 Technical assistance

For assistance or clarification on information in this document, submit a case to Qualcomm Technologies, Inc. (QTI) at https://createpoint.qti.qualcomm.com/.

If you do not have access to the CDMATech Support website, register for access or send email to support.cdmatech@qti.qualcomm.com.

2 Installation

2.1 Required hardware and software

QXDM Pro is designed to be installed and run on workstations running Windows 10 operating systems. QXDM Pro requires Microsoft Internet Explorer 6.0 or later and QUTS 0.2.38 or later. Both QXDM and QUTS can be installed via Qualcomm Package Manager.

Minimum system requirements

The minimum required configuration for QXDM Pro is described in Table 2-1.

The extensibility features of QXDM Pro enable it to support unlimited configurations including support for end user-defined applications. A minimum configuration necessarily means that less work can be done simultaneously. Multiple instances of QXDM Pro, other running (including background) processes, and having too many views open at the same time may require more than the minimum system configuration listed below.

Table 2-1 Minimum system requirements

Item	Description
CPU	800 MHz Pentium III class
RAM	1 GB
Hard drive	Installation requires 600 MB. Logging requires minimum 20 MB free disk space. Note that some of today's phones are capable of generating log data in excess of 18 MB per minute.
Operating systems	Windows 10
Port connections	One USB or serial I/O port per device (phone, FFA, SURF, or GPS receiver)
Software requirements	Qt5WebKit.dll

2.2 Installing QXDM Pro

The installer sets up the QXDM Pro execution environment, which includes installing application binaries, data files, and documentation; registering COM automation components and file associations; and configuring QXDM Pro for initial use.

The QXDM Pro installation consists of two main folders:

- **QXDM program folder** – The path to this folder is set by the user when installing QXDM Pro. The default path offered by the installer is based on the underlying Microsoft operating system program files folder, which is typically C:\Program Files (x86)\Qualcomm\ QXDM5. After installation, this folder will contain subfolders containing the QXDM Pro binaries, and QXDM Pro documentation. Under this will be several subfolders containing QXDM Pro automation script samples, implementation files for all QXDM Pro HTML-based displays.
- **QXDM data folder** – The base path to this folder is set by the underlying Microsoft operating system and represents the documents folder shared by all users of the host PC. Typically, the Microsoft Windows shared documents folder is located at C:\Program Files (x86)\Qualcomm\ Shared\.

  The resulting complete path represents the QXDM data folder. Under this will be several subfolders containing the different QXDM Pro databases related with different technologies.

  It also contains a folder named C:\ProgramData\QUALCOMM\QXDM\Config\Qualcomm DMC Library\Primary, which includes the default DMC file.

The folder located in C:\users\username\AppData\Local\QXDM contains DMC files, user database (the default path can be changed), and other supporting files. The installer creates a QXDM Pro folder in the Windows Start Programs menu that can be run by selecting Start > All Programs > QXDM. The installed application binaries and user guides are accessible from this location.

2.2.1 QPM license activation

A one-time online activation is required before QXDM Pro can be run using Qualcomm Package Manager. Follow the instructions provided by QTI to activate QXDM Pro.

2.3 Physical connectivity

QXDM Pro connects to a phone or SURF through the QUTS using a serial or USB cable to your PC.

2.4 QXDM Pro documentation

QXDM supporting documents are available at https://createpoint.qti.qualcomm.com/.

- **Release Notes** – Revision changes for each version of QXDM Pro are detailed in the ReleaseNotes file as part of the installation package.

2.5 Item store format (.hdf) files

While running, QXDM Pro generates a temporary log file called the Item Store, which ensures that data is not lost in the event of unexpected program termination. Under normal circumstances, this file is deleted upon termination.

The Item Store contains all traffic that occurs between QXDM Pro and the target (or targets if QXDM Pro sequentially connects to more than one target in a given session) or since the last time the Item Store was cleared, e.g., via the Save Item (**Ctrl + I**) menu command. This provides an accurate representation of the state of the application at any given time for analysis and debugging.

Always-On logging

QXDM Pro is always generating an HDF log file. Log files can be processed by QXDM Pro at a later time for post processing analysis and replay. Always-On logging behavior can be customized from the Item Store Settings command.

The Application Statistics view is the place to go for monitoring the status of logging.

Recovering the Item Store

If for some reason QXDM Pro is terminated abnormally, the session can be recovered. Section 3.1 describes how to load an HDF file. In the event of abnormal termination, it may be possible to recover and repair the lost session. The temporary HDF is located in the C:\Windows\Temp\QUTS.

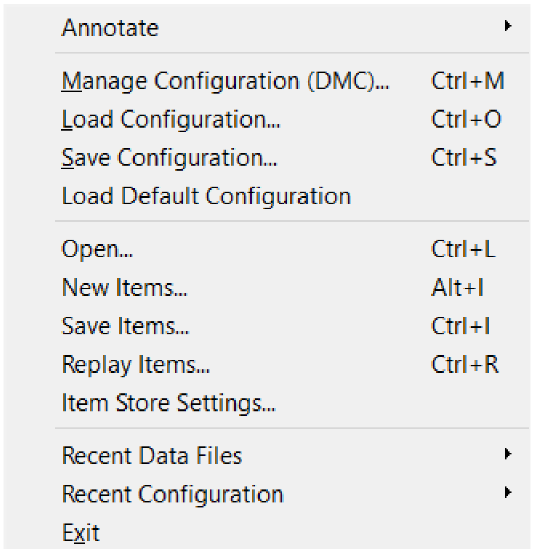
Saving the Item Store

By default, the HDF is deleted upon QXDM Pro termination. To save a session for later analysis or viewing, select **Enable Query For HDF Save** (see Figure 3-1).

3 Menu overview

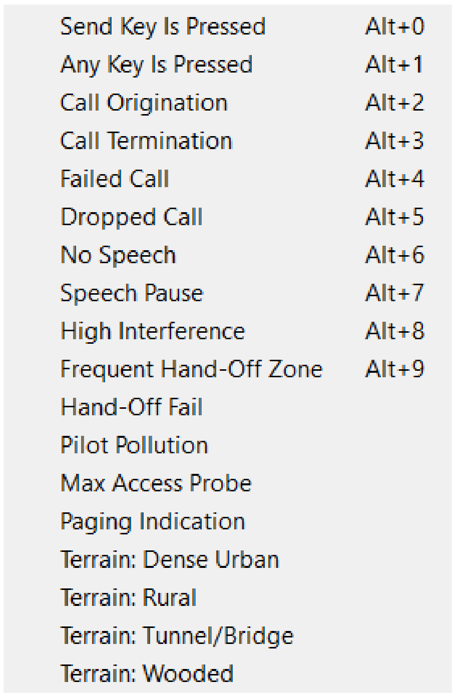
QXDM Pro uses a multiple document interface to support any number of views simultaneously.

3.1 File menu

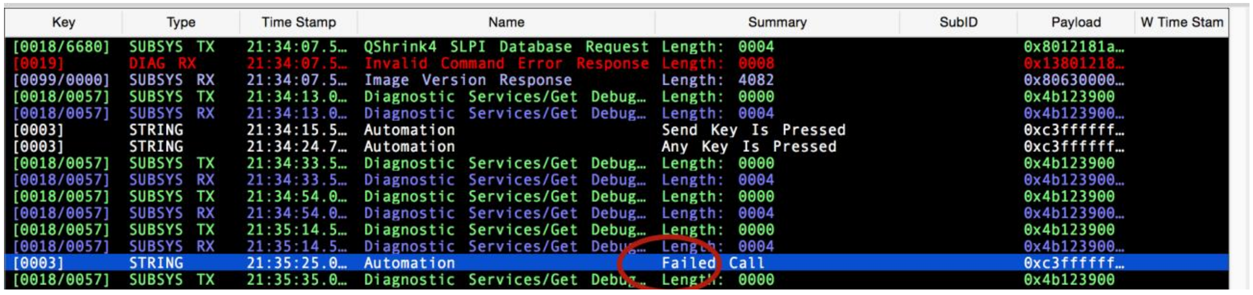


3.1.1 Annotation

User can insert annotation in the item view at real-time while gathering log. The added annotation will be saved as part of the .hdf file. This functionality is only available during log collection phase, and will not work in an already saved .hdf file.



Sample:



```
Key           Type        Time Stamp    Name                             Summary               SubID   Payload        W Time Stam
[0018/6680]   SUBSYS TX   21:34:07.5…   QShrink4 SLPI Database Request   Length: 0004                  0x8012181a…
[0019]        DIAG RX     21:34:07.5…   Invalid Command Error Response   Length: 0008                  0x13801218…
[0099/0000]   SUBSYS RX   21:34:07.5…   Image Version Response           Length: 4082                  0x80630000…
[0018/0057]   SUBSYS TX   21:34:13.0…   Diagnostic Services/Get Debug…   Length: 0000                  0x4b123900
[0018/0057]   SUBSYS RX   21:34:13.0…   Diagnostic Services/Get Debug…   Length: 0004                  0x4b123900…
[0003]        STRING      21:34:15.5…   Automation                       Send Key Is Pressed           0xc3ffffff…
[0003]        STRING      21:34:24.7…   Automation                       Any Key Is Pressed            0xc3ffffff…
[0018/0057]   SUBSYS TX   21:34:33.5…   Diagnostic Services/Get Debug…   Length: 0000                  0x4b123900
[0018/0057]   SUBSYS RX   21:34:33.5…   Diagnostic Services/Get Debug…   Length: 0004                  0x4b123900…
[0018/0057]   SUBSYS TX   21:34:54.0…   Diagnostic Services/Get Debug…   Length: 0000                  0x4b123900
[0018/0057]   SUBSYS RX   21:34:54.0…   Diagnostic Services/Get Debug…   Length: 0004                  0x4b123900…
[0018/0057]   SUBSYS TX   21:35:14.5…   Diagnostic Services/Get Debug…   Length: 0000                  0x4b123900
[0018/0057]   SUBSYS RX   21:35:14.5…   Diagnostic Services/Get Debug…   Length: 0004                  0x4b123900…
[0003]        STRING      21:35:25.0…   Automation                       Failed Call                   0xc3ffffff…
[0018/0057]   SUBSYS TX   21:35:35.0…   Diagnostic Services/Get Debug…   Length: 0000                  0x4b123900
```

3.1.2 Manage Configuration

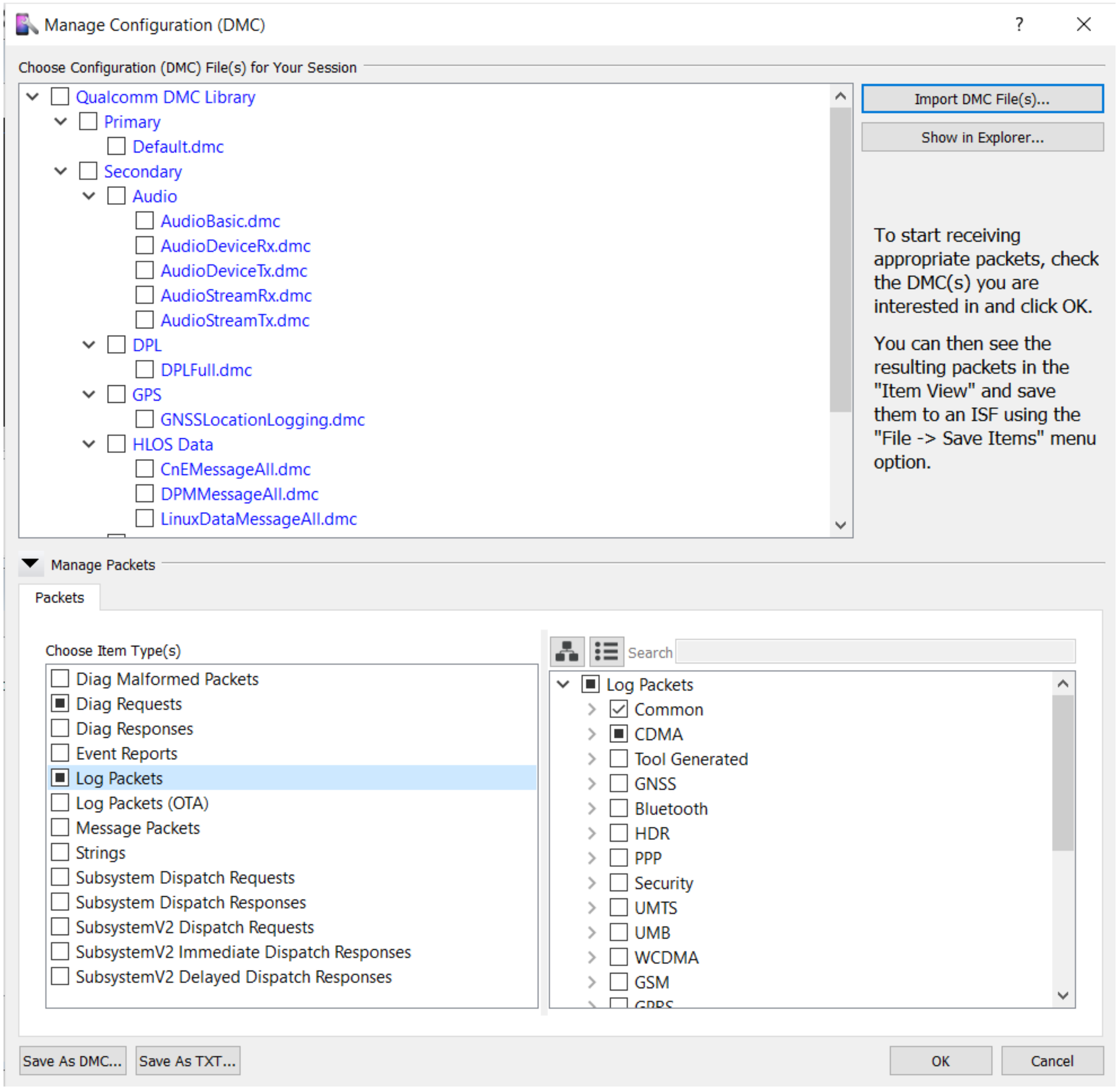
Manage configuration with diagnostic monitor configuration (DMC)

Use DMC files to set the data packets to be logged. Setting the correct DMC ensures that correct packets are collected. DMCs can also specify which views need opening for further analysis.

The Manage Configuration window shows QTI and user DMCs. Select the desired DMCs and click **OK**. The DMC selection sends the request for the packets, as shown in the Item View window.

Click **Import DMC File(s)** to add additional DMCs, or click **Show in Explorer** to manage DMCs within the file explorer.

Click **Save As DMC** to save the selected DMCs as a single DMC, or click **Save As TXT** to save the packets as text.



To start receiving appropriate packets, check the DMC(s) you are interested in and click OK.

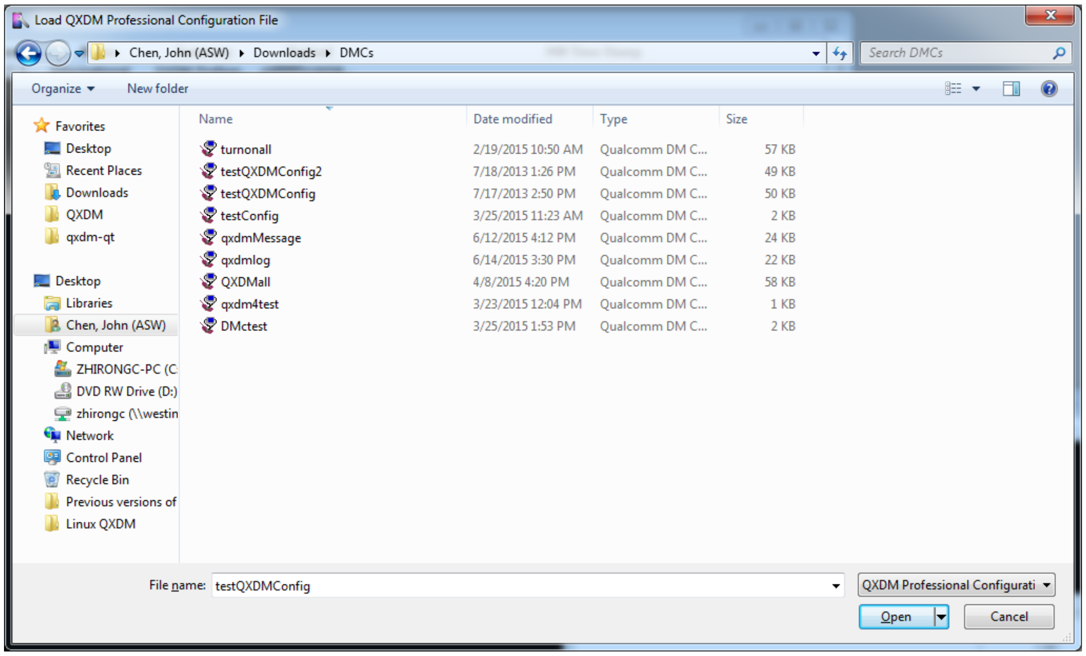
You can then see the resulting packets in the "Item View" and save them to an ISF using the "File -> Save Items" menu option.

- [ ] Diag Malformed Packets
- [■] Diag Requests
- [ ] Diag Responses
- [ ] Event Reports
- [■] Log Packets
- [ ] Log Packets (OTA)
- [ ] Message Packets
- [ ] Strings
- [ ] Subsystem Dispatch Requests
- [ ] Subsystem Dispatch Responses
- [ ] SubsystemV2 Dispatch Requests
- [ ] SubsystemV2 Immediate Dispatch Responses
- [ ] SubsystemV2 Delayed Dispatch Responses

- [■] Log Packets
  - [✓] Common
  - [■] CDMA
  - [ ] Tool Generated
  - [ ] GNSS
  - [ ] Bluetooth
  - [ ] HDR
  - [ ] PPP
  - [ ] Security
  - [ ] UMTS
  - [ ] UMB
  - [ ] WCDMA
  - [ ] GSM
  - [ ] GPRS

3.1.3 Load Configuration

Previously saved configurations can be loaded using the Load Configuration command. Configurations such as selected views and registrations are restored from QXDM Pro configuration files (.dmc extension).

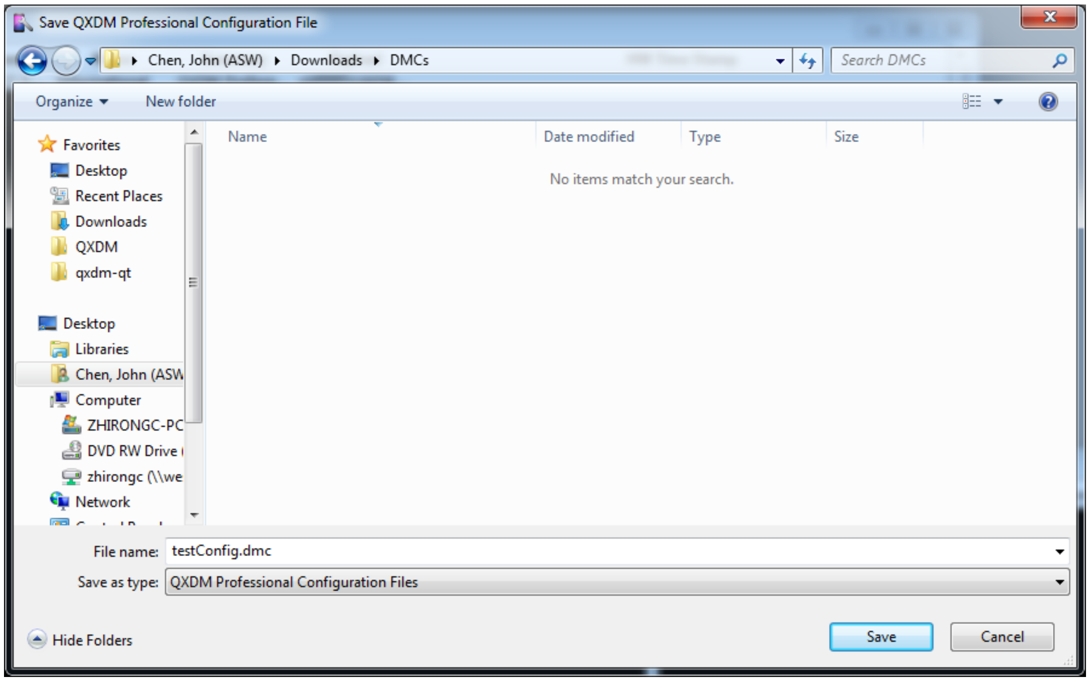


Name	Date modified	Type	Size
turnonall	2/19/2015 10:50 AM	Qualcomm DM C...	57 KB
testQXDMConfig2	7/18/2013 1:26 PM	Qualcomm DM C...	49 KB
testQXDMConfig	7/17/2013 2:50 PM	Qualcomm DM C...	50 KB
testConfig	3/25/2015 11:23 AM	Qualcomm DM C...	2 KB
qxdmMessage	6/12/2015 4:12 PM	Qualcomm DM C...	24 KB
qxdmlog	6/14/2015 3:30 PM	Qualcomm DM C...	22 KB
QXDMall	4/8/2015 4:20 PM	Qualcomm DM C...	58 KB
qxdm4test	3/23/2015 12:04 PM	Qualcomm DM C...	1 KB
DMctest	3/25/2015 1:53 PM	Qualcomm DM C...	2 KB

File name: testQXDMConfig

3.1.4 Save configuration

Configurations remember selected view registrations and settings and can be saved to QXDM Pro configuration files (.dmc extension) for later loading.



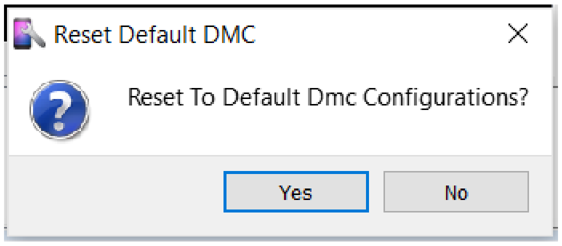
No items match your search.

File name: testConfig.dmc

Save as type: QXDM Professional Configuration Files

3.1.5 Load Default Configuration

Click “Yes” will restore default dmc and views



Reset To Default Dmc Configurations?

3.1.6 Open

Select a QXDM supported file for post processing and modification. If there are filter views open, opening a .HDF file would populate the filter views based on the log masks. Therefore, expects delay if opening a large .HDF file with filter views open.

3.1.7 New Items

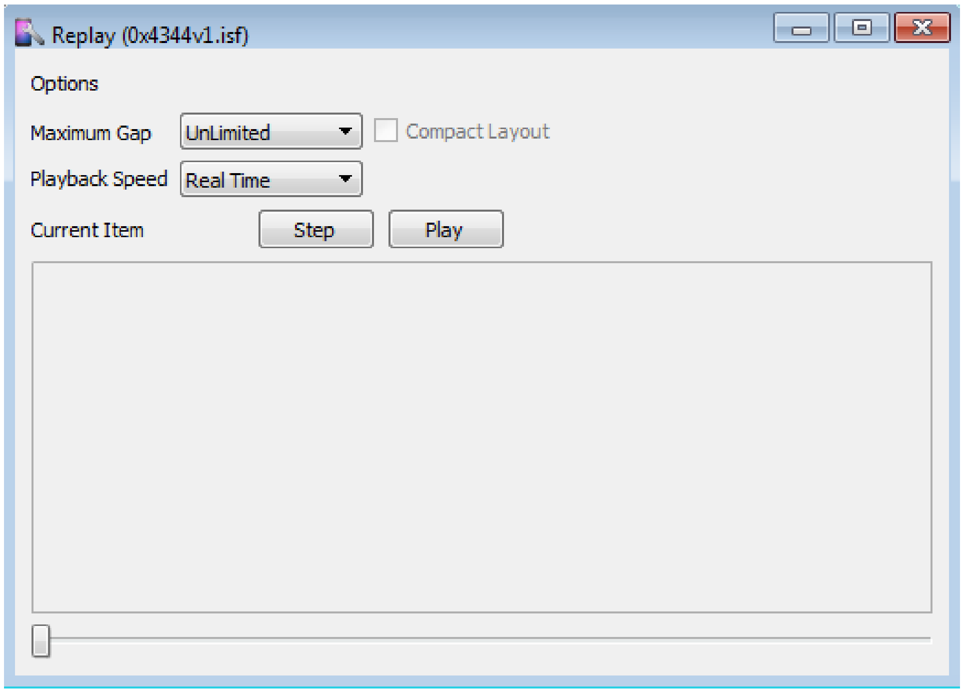
New Items creates a new temporary .hdf file. This is useful if you have been reviewing an existing file store (by loading or replaying it) and now want to connect to a live target without appending to the current file. This option can also be used to start a new .hdf file in advance of commencing a test.

3.1.8 Save Items

Save Items behaves exactly the same as New Items except that there is no check to see if Enable Query For HDF Save is checked.

3.1.9 Replay Items

A previously saved .hdf file can be loaded for replaying using this command.



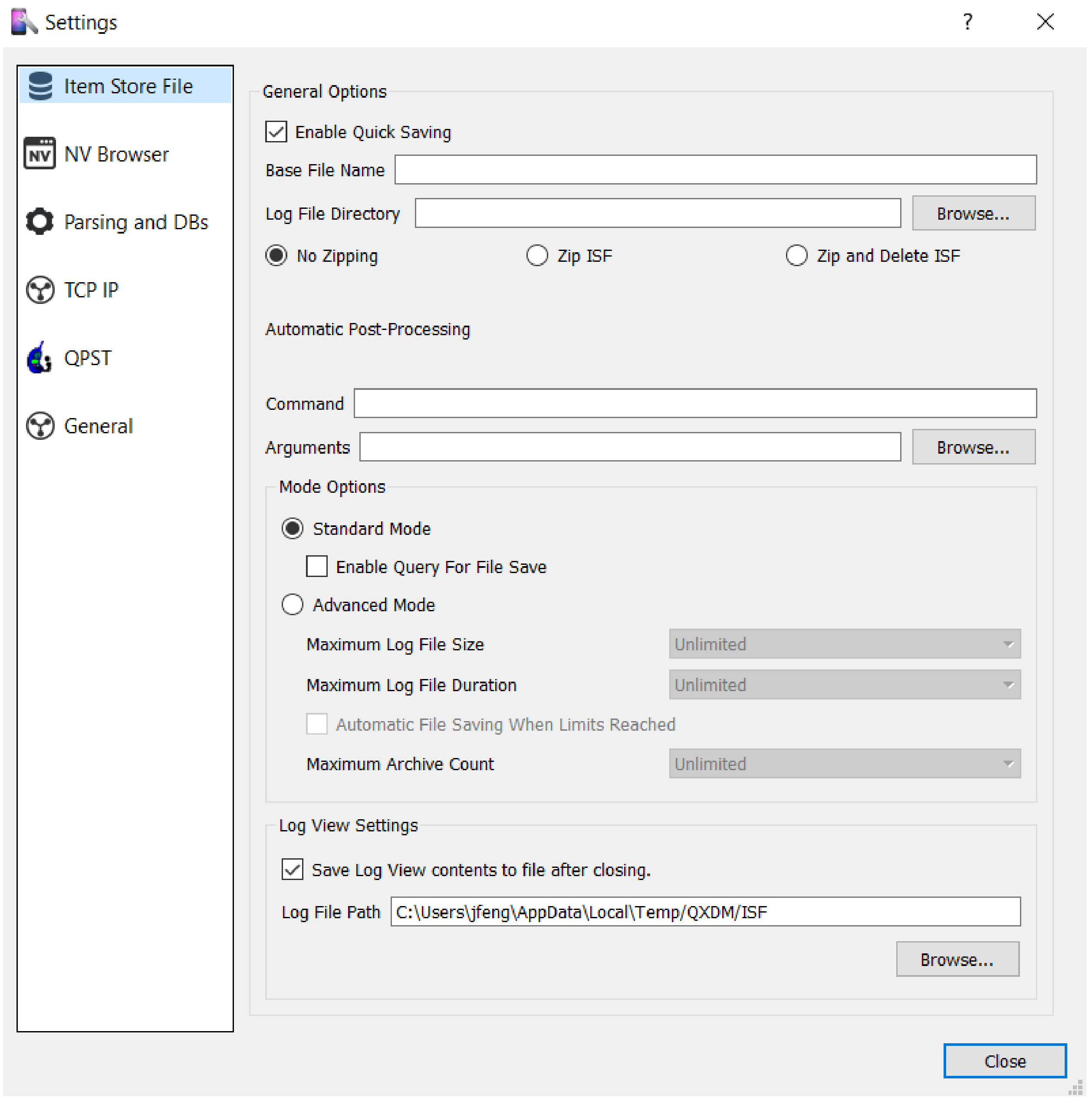
- **Maximum Gap** – Set the maximum amount of time in milliseconds to wait before playing the next item. If the actual gap exceeds the configured value, the configured value will be used as the amount of time to wait until playing the next item.
- **Playback Speed** – Set the speed at which items are played.

NOTE: Using the No Wait option can impact system performance when views are open that process each and every item.

- **Compact Layout** – Information about an item is displayed as it is processed unless Compact Layout is checked.
- **Step** – Replay just one item
- **Play/Pause** – Replay based on the selected settings. To stop replaying, click **Pause**.

3.1.10 Item Store Settings

The Item Store Settings dialog allows the user to control HDF logging behavior.


Settings

General Options

[✓] Enable Quick Saving

Base File Name:

Log File Directory: Browse...

[● No Zipping] ○ Zip ISF ○ Zip and Delete ISF

Automatic Post-Processing

Command:

Arguments: Browse...

Mode Options

[● Standard Mode]

[ ] Enable Query For File Save

○ Advanced Mode

Maximum Log File Size: Unlimited

Maximum Log File Duration: Unlimited

[ ] Automatic File Saving When Limits Reached

Maximum Archive Count: Unlimited

Log View Settings

[✓] Save Log View contents to file after closing.

Log File Path: C:\Users\jfeng\AppData\Local\Temp/QXDM/ISF

Figure 3-1 Item Store File Settings

3.1.10.1 Item Store File

The default behavior is for QXDM Pro to prompt for a log file name on user-initiated log file saves (e.g., File→Save Items).

- **Enable Quick-Saving** – When Enable HDF Quick-Saving is checked, the log file is saved using the Base HDF Name at the HDF Directory location, without prompting the user for a filename.

NOTE: This applies only to scenarios where the user has initiated the save, i.e., by selecting the **Save Items** menu command. In the other scenarios, where an HDF must be saved due to a parallel operation (New Items, Load Items, and exiting QXDM), the user may still be prompted for a filename (dependent on the mode). In all scenarios, an automatically generated name will appear in the prompt dialog itself.

  - **Base Folder Name** – Use Base HDF Name to describe the unique portion of the HDF foldname that will be used when in Advanced Mode. Using the base HDF name, QXDM Pro will format the name as follows:

    ```
    <Base Name>MM-DD-HH-MM.HDF
    ```

  - **Log File Directory** – Use HDF Directory to describe where to save HDF files. The default is located in the Microsoft Windows shared documents folder for all users typically at C:\Documents and Settings\All Users\Documents\Qualcomm\QXDM\HDF.

- **Automatic Post-Processing** – Automatic post processing of log files is supported by providing a command and optional arguments that QXDM Pro will run each time a log is saved. The log is passed to the specified command by QXDM Pro.
  - **Command** – Enter the command including the fully qualified path that QXDM Pro is to run, e.g., C:\WINDOWS\System32\CScript.exe.
  - **Arguments** – Enter any required arguments for the command, e.g., C:\TestScripts\HDFAnalyzeLog1234.js. The format requirements are the same as if run from a command window, i.e., if the argument contains spaces, it should be enclosed in quotes.
  - **Sort saved HDF file on timestamp**: 1. Enter in Commands: *python*, 2. Enter in Arguments: *C:\ProgramData\Qualcomm\QXDM\postProcessingScripts\open_file_to_sort_it_after_saving.py*. The setting will be saved to DMC when exiting QXDM.

Mode Options

The default logging mode is for QXDM Pro to delete the temporary log file upon program termination.

- **Standard Mode** – Selecting Standard Mode results in the default behavior described above.
  - **Enable Query For File Save** – When this option is checked in Standard Mode, the user is prompted to save the temporary Item Store upon creating a new Item Store (New Items), loading an Item Store (Load Items), or before exiting.

- **Advanced Mode** – Selecting this option causes QXDM Pro to apply the advanced options described below.
  - **Maximum Log File Size** – Use Maximum HDF Size to set the maximum file size the current temporary HDF can reach before it is closed and a new temporary HDF is used.
  - **Maximum Log File Duration** – Use Maximum HDF Duration to control the maximum file duration (time between first item's generic timestamp and last item's generic timestamp) that can elapse before the current temporary HDF is closed and a new HDF is used.
  - **Automatic File Saving When Limits Reached** – This checkbox controls whether an HDF that has exceeded specified limits is automatically saved. If it is not checked, the temporary HDF is deleted when limits are reached.
  - **Maximum Archive Count** – Use Maximum Archive Count to control the maximum number of HDF files that will be saved. When this value is exceeded, the oldest HDF saved by the current QXDM Pro process is deleted.

Log View Settings

- Save Log View content to file after closing – If selected, all content in the item views will be saved.
- Log File Path – File path for saving log view content.

3.1.10.2 Parsing and DBs

Protocol Categories/Revisions

- **CDMA** – This is used to control the protocol revision utilized by QXDM when parsing CDMA over-the-air (OTA) messages. Item Store files created by QXDM version 03.08.99 or later utilize the most recent CDMA protocol revision reported by the target. For these files (and real-time interaction), the Automatic choice is the best selection. For files created by QXDM versions prior to 03.08.99 the option should be set to the appropriate protocol revision.

  Utilizing the Automatic choice when viewing an HDF created by QXDM versions prior to 03.08.99 is equivalent to choosing [06] – IS-2000 Rev 0.
- **WCDMA** – This is used to control the protocol revision utilized by QXDM when parsing WCDMA OTA messages. Item Store files created by QXDM version 03.09.15 or later utilize the WCDMA protocol revision reported by the target. For these files (and real-time interaction), the Automatic choice is the best selection. For files created by QXDM versions prior to 03.09.15, the option should be set to the appropriate protocol revision.

  Utilizing the Automatic choice when viewing an HDF created by QXDM versions prior to 03.09.15 is equivalent to choosing 06/04 V5.9.0.
- **HSDPA** – This setting allows control of the HSDPA UE category used when the QXDM HSDPA displays need to map CQI indices to bit sizes. This value should match the UE category of the connected target.
- Vendor Database Path
  - A vendor-specific database is used when parsing items specific to a particular handset vendor. A vendor-specific database can also contain mobile model names.

- QShrink3 User Hash Location
  - Location of QShrink3 database
- QShrink4 Server Path
  - Server location of QShrink4 database.

3.1.10.3 General

- Font For Item Lists: Change the font of the item list view
- Default Plot Legend Area (%): Set the default size of legend window for all graphic views.
- Plot Background Color: Set the default background color for all graphic views.

3.1.11 Recent Data File

Display the history of past loaded item store files

3.1.12 Recent Configuration

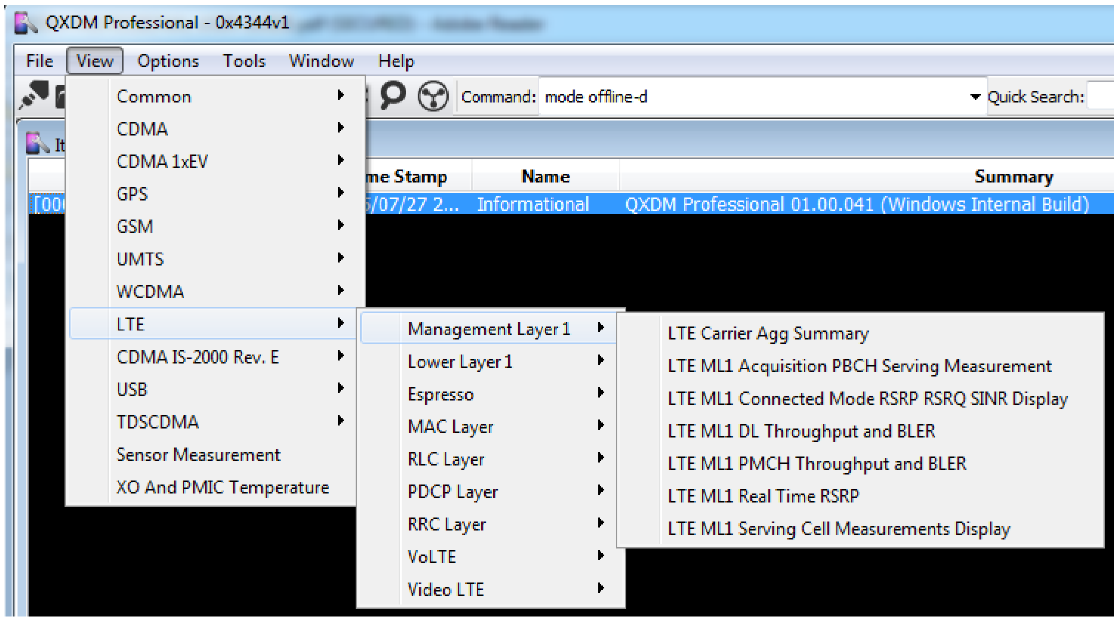
Display the history of past loaded configuration files

3.1.13 Exit

Click **Exit** to end the QXDM Pro session.

3.2 View menu

This section discusses options available through the View menu selection.



me Stamp	Name	Summary
/07/27 2...	Informational	QXDM Professional 01.00.041 (Windows Internal Build)

Views under view menu are created based on diagnostic packet definition. Tables, charts, and values are displayed based on diagnostic packet received from UE. For descriptions and details of diagnostics packets, please reference to technology interface control documents (ICD) This section only covers a few commonly used views.Item & Filter Views

An item view is a scrolling view that contains zero or more items that are derived from the contents of the current .HDF log file. This includes the Item View, Filtered Views, the Messages View, the Log View, Command Item View and the Command Output display.

3.2.1 Item View

The Item View is a special item list view that shows all items generated during a QXDM Pro session. When QXDM Pro is run, a temporary HDF is created in the QXDM Pro HDF folder (typically located at C:\Users\user\AppData\Local\Temp\QXDM\HDF). The Item View shows the contents of this temporary HDF log file and can be displayed from the View bar or by using the **F11** accelerator key.

An item list view has three adjustable panes. The top pane is a scrolling list where a summary of each item in the view is displayed. The bottom left pane is used to display (in a hexadecimal dump) the raw data of the item currently selected in the scrolling list. The bottom right pane is used to display the parsed fields or (when available) the parsed text of the currently selected item in the scrolling list.

3.2.2 Filtered View

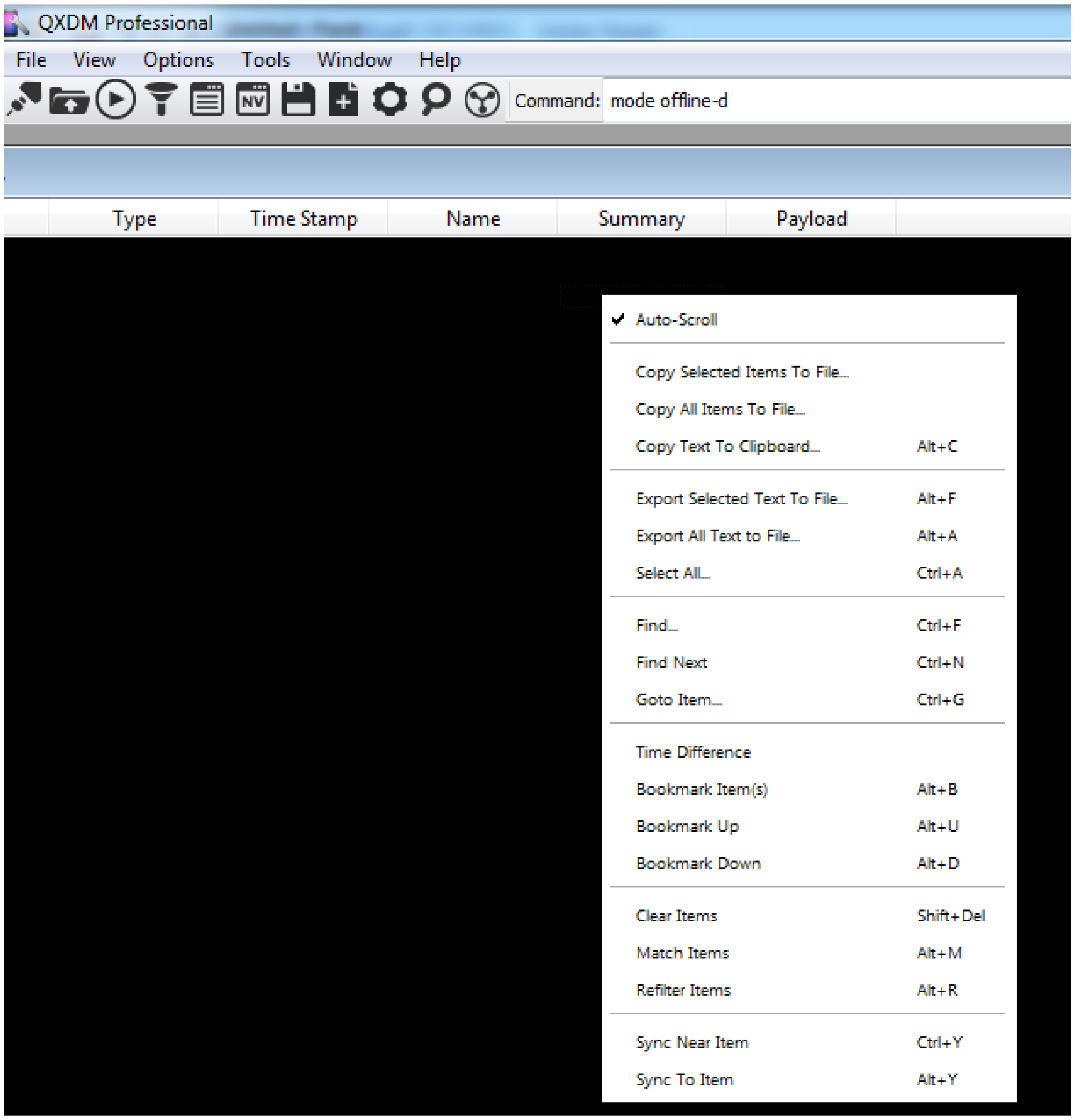
A Filtered View represents a subset of the contents of the current .HDF file and therefore the Item View. This subset is configured by item type and/or item key. Unlimited filtered views may be created by using the accelerator key **F12** or selecting Filtered View from the View Bar. Filtered views can also be created by interacting with an existing item list based display through the Refilter Items, Match Items, and Process Items menu commands.

3.2.2.1 Scrolling list pane

All item list-based QXDM Pro displays contain the scrolling list pane. This includes the Item View and Filtered Views. This pane displays each item in the view, one per line in the list.

Right-Click options

Right-clicking within the scrolling list pane brings up the context-sensitive menu.



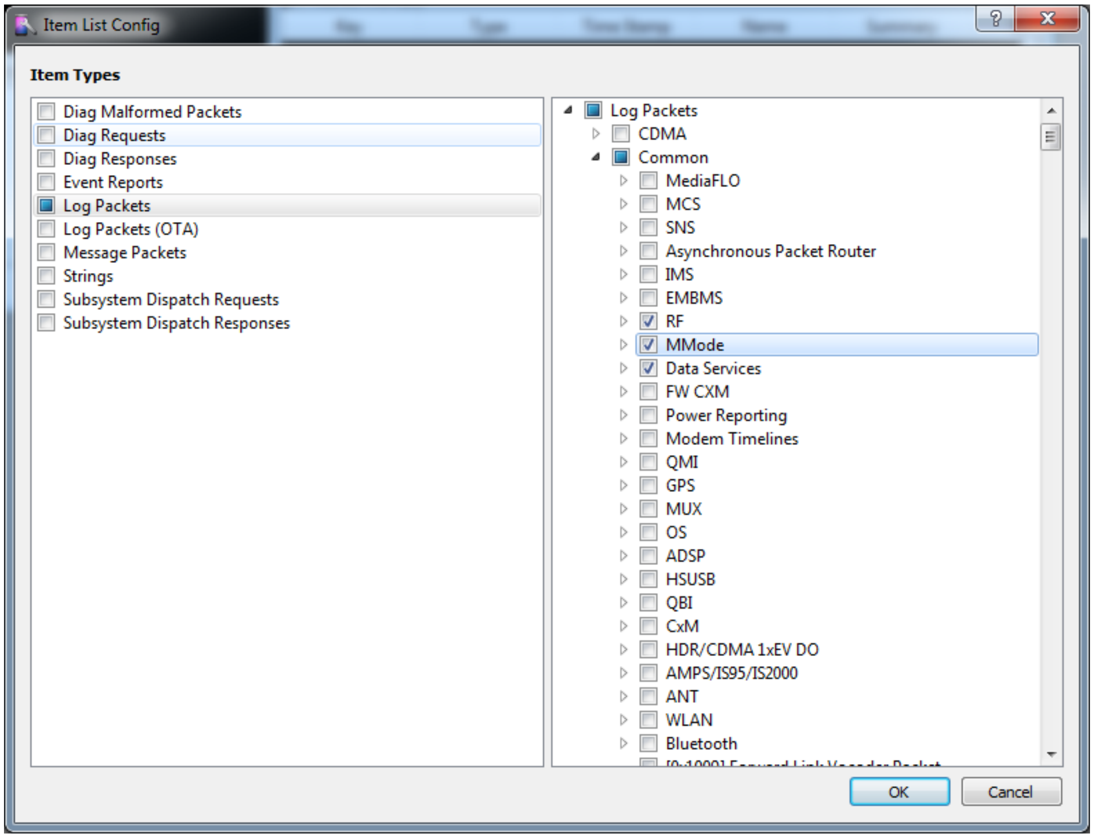
The context-sensitive menu is used to configure view registration, alter appearance, copy text, locate items, clear the view, and view raw item content.

Configuration in Filter View

Selecting this menu option allows control of which items are added to the view in question.

The left side of the configuration dialog is used to identify what item types the filtered view will accept. The right side of the dialog is used to filter item type content based on item key and to register with the phone for items of interest.

This configuration dialog is only supported by Filtered Views.



Items are grouped into types, as shown in Table 4-1. Item types allow QXDM Pro to classify incoming data from multiple sources: synchronous and asynchronous traffic between QXDM Pro and a target, GPS receiver data, and internally generated strings

Table 4-1 Item types

Item type	Source
Diag Malformed Packets	Phone
Diag Requests	QXDM, User
Diag Responses	Phone
Event Reports	Phone
GPS Reports	GPS Receiver
Log Packets	Phone
Log Packets (OTA)	Phone
Message Packets	Phone
Strings	QXDM, User
Subsystem Dispatch Requests	QXDM, User
Subsystem Dispatch Responses	Phone

Copying

All or part of a filtered view output can be copied to a new HDF, a text file, or to the clipboard. The action is controlled by using any of the methods given in Table 4-2.

Table 4-2 Copying items

Item	Description
Copy Items	Copy selected items to an Item Store Format (.HDF) file
Copy All Items	Copy All Items to an Item Store Format (.HDF) file
Copy Text (**Alt** + **C**)	Copy selected text to the clipboard
Copy Head Line Only (**Ctrl + C**)	Copy head line from an item
Export Text (**Alt** + **F**)	Export selected items as text to a file
Export All Text (**Alt** + **A**)	Export all items as text to a file
Select All (**Ctrl** + **A**)	Select all items

Text output from loaded dynamic parsers, built-in OTA parsers, and extended database descriptions will also be copied if the menu item Options→Export Parsed Text is selected.

Text output is subject to change. Use postprocessing tools, e.g., QCAT, to generate consistent text output for more reliable text processing results.

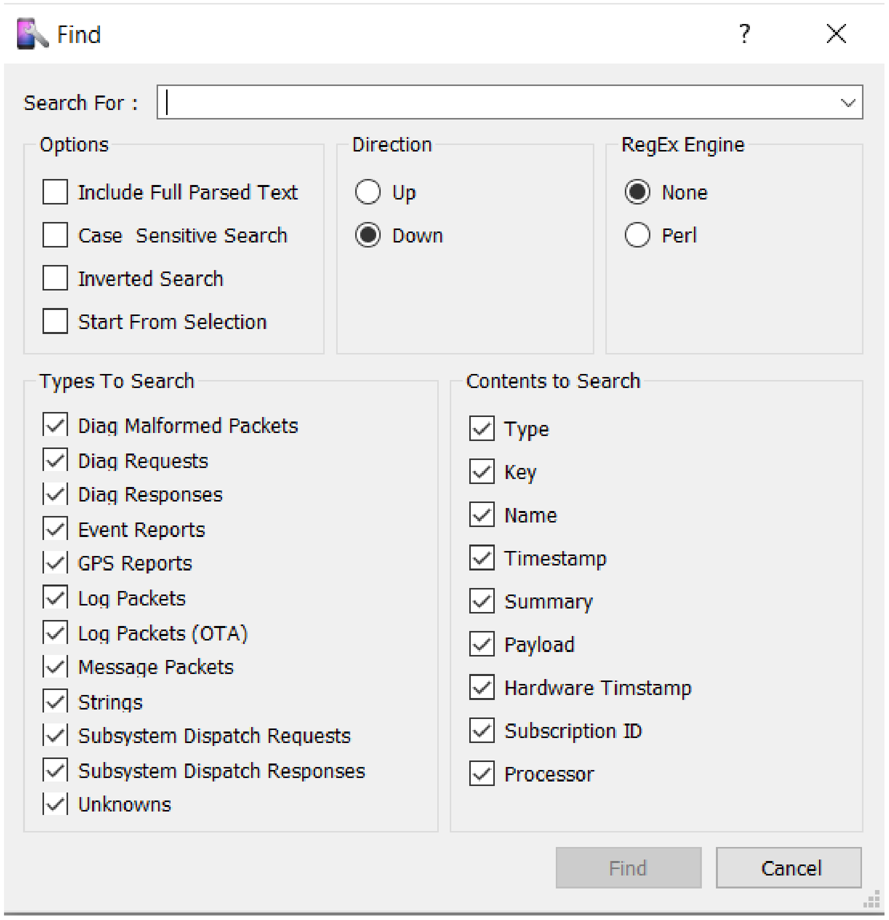
All of the above operations, excluding copying text to the clipboard, result in the display of a progress dialog. This dialog displays the progress of the operation, statistics related to the operation, and any errors encountered in the process of the operation (such as errors reading an item from the current HDF or writing an item out to the new HDF). Additionally, the progress dialog allows for cancellation of the operation before it has completed. In this case, the partial results are written out to the new HDF or text file.

Searching

Each line in an item list view is the text representation of the properties of a particular item. This text can be searched to find an item that meets given search criteria. Additionally, each item in an item list view is associated with an index. This index can be used to jump to an item. For items with type Message Packets, additional information present in the content of the item can be used to locate the point in the target source code where the item originated.

Find

Selecting this menu option (or using the **Ctrl** + **F** keyboard shortcut) allows searching for text within the item list output. Both the type of item and the item properties to be searched can be configured.


Find

Search For :

Options
- [ ] Include Full Parsed Text
- [ ] Case Sensitive Search
- [ ] Inverted Search
- [ ] Start From Selection

Direction
Up [✓ Down]

RegEx Engine
[✓ None] Perl

Types To Search
- [✓] Diag Malformed Packets
- [✓] Diag Requests
- [✓] Diag Responses
- [✓] Event Reports
- [✓] GPS Reports
- [✓] Log Packets
- [✓] Log Packets (OTA)
- [✓] Message Packets
- [✓] Strings
- [✓] Subsystem Dispatch Requests
- [✓] Subsystem Dispatch Responses
- [✓] Unknowns

Contents to Search
- [✓] Type
- [✓] Key
- [✓] Name
- [✓] Timestamp
- [✓] Summary
- [✓] Payload
- [✓] Hardware Timstamp
- [✓] Subscription ID
- [✓] Processor

In addition to search direction, the following options are supported:

- **Include Full Parsed Text (When Available)** – When this is selected, the search engine will also search the full parsed text that appears in the lower right-hand pane of the view.
- **RegEx Engine** – The parser will accept regular expressions when JScript or Perl options are selected. Although support for JScript is included with the Windows operating system, Perl requires a separate installation. Documentation on regular expression syntax for these scripting languages is beyond the scope of this document.
- **Case Sensitive Search** – By default, the search is case-insensitive.

- **Inverted Search** – When this is selected, the search engine will search for any pattern not matching the search criteria from the set of item types and item properties.
- **Start From Last Selection** – By default, a new search is from the beginning of the file. Checking **Start from Last Selection** overrides this behavior.

Other search options include:

- **Find Next** – Search for the next Find match. This command can also be executed by pressing **Ctrl** + **N**.
- **Goto Item** – Go to an item by specifying its index using the Goto Item command. When successful, QXDM Pro will highlight the item at the index specified. This command can be executed by pressing **Ctrl** + **G**.
- **View Source** – View the target source file associated with a selected item. This command can also be executed by pressing **Alt** + **V**. Use **Options** → **Settings** (see Section 3.3.2) to specify the search path used by QXDM Pro.
- **Auto-Scroll** – When new items arrive and are added to an item list view, QXDM Pro can automatically scroll to the end of the item list in order to display the new items, The Auto-Scroll menu option allows for enablement or disablement of this behavior. Control automatic scrolling by pressing **Alt** + **S** or by manually scrolling using the scroll bar, arrows, or **Page Up** and **Page Down** keys.

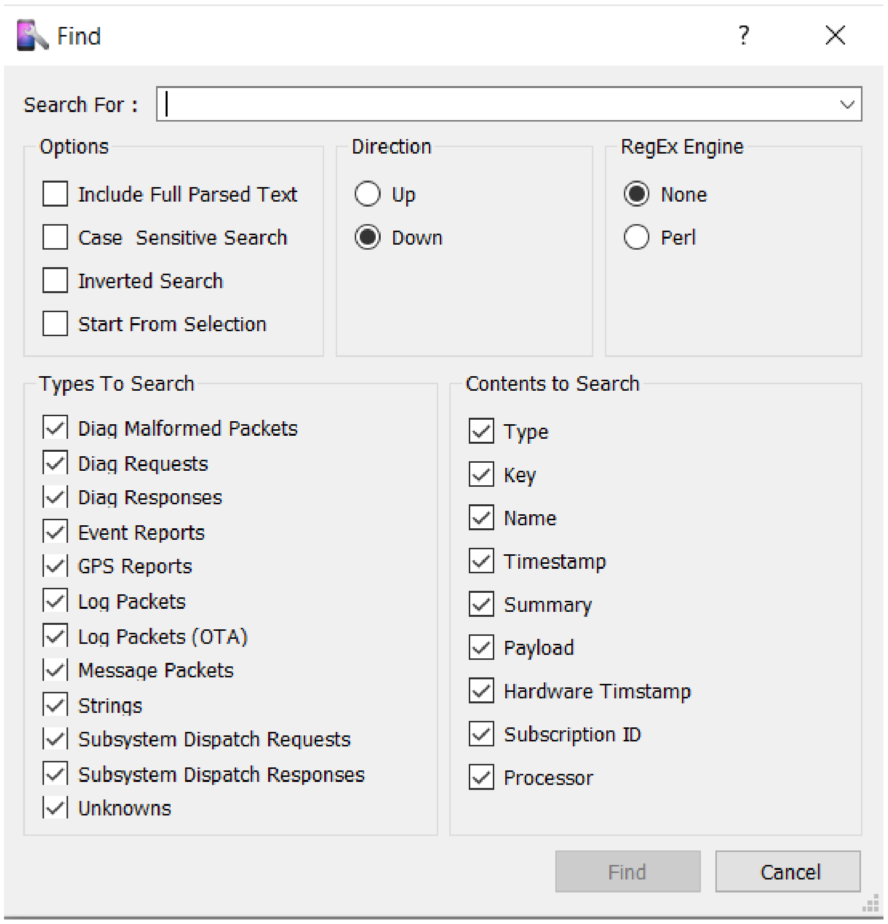
Clear Items

Clear Items allows all items to be flushed from an item list view. Items may also be cleared by pressing **Shift** + **Delete**.

Items can be cleared from any scrolling view except the Logging <**Alt** + **L**> View. Take care when clearing the Item View because doing so results in the current HDF being emptied and the connection being reset.

Match Items

Match Items allows for creation of a new filtered view based on a matched pattern from a selected set of items in an existing item list view. Configuring this process is nearly the same as configuring a find operation.



- [ ] Include Full Parsed Text
- [ ] Case Sensitive Search
- [ ] Inverted Search
- [ ] Start From Selection

Direction: Up [✓ Down]

RegEx Engine: [✓ None] Perl

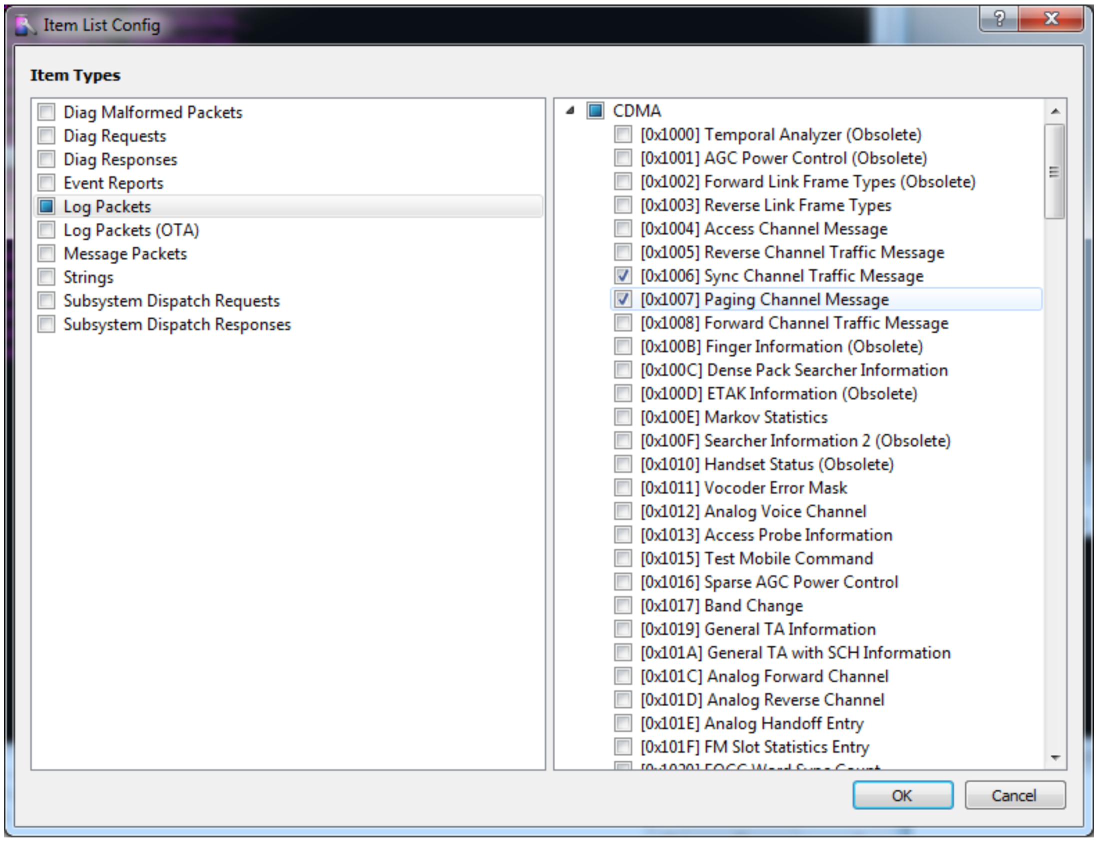
Types To Search:
- [x] Diag Malformed Packets
- [x] Diag Requests
- [x] Diag Responses
- [x] Event Reports
- [x] GPS Reports
- [x] Log Packets
- [x] Log Packets (OTA)
- [x] Message Packets
- [x] Strings
- [x] Subsystem Dispatch Requests
- [x] Subsystem Dispatch Responses
- [x] Unknowns

Contents to Search:
- [x] Type
- [x] Key
- [x] Name
- [x] Timestamp
- [x] Summary
- [x] Payload
- [x] Hardware Timstamp
- [x] Subscription ID
- [x] Processor

The only difference is that the direction and selection options are not available. Instead, the operation starts from the first selected item in the current item list view and ends with the last selected item.

Process Items

- **Refilter Items** – Create a new filtered view from a selected group of items in an existing view based on item type and/or item key. Configuring the refilter operation is no different than configuring a Filtered View.

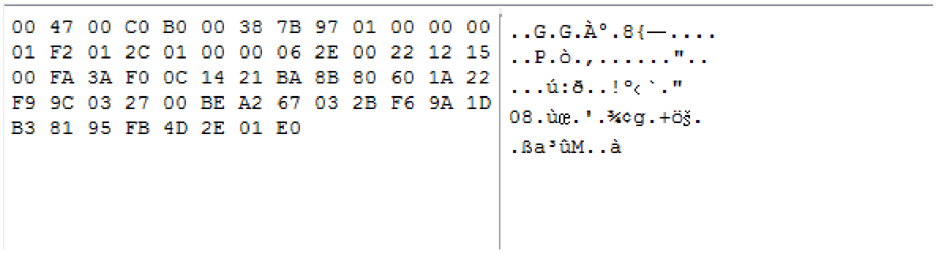


Item Types

- ☐ Diag Malformed Packets
- ☐ Diag Requests
- ☐ Diag Responses
- ☐ Event Reports
- ■ Log Packets
- ☐ Log Packets (OTA)
- ☐ Message Packets
- ☐ Strings
- ☐ Subsystem Dispatch Requests
- ☐ Subsystem Dispatch Responses

■ CDMA
- ☐ [0x1000] Temporal Analyzer (Obsolete)
- ☐ [0x1001] AGC Power Control (Obsolete)
- ☐ [0x1002] Forward Link Frame Types (Obsolete)
- ☐ [0x1003] Reverse Link Frame Types
- ☐ [0x1004] Access Channel Message
- ☐ [0x1005] Reverse Channel Traffic Message
- ✓ [0x1006] Sync Channel Traffic Message
- ✓ [0x1007] Paging Channel Message
- ☐ [0x1008] Forward Channel Traffic Message
- ☐ [0x100B] Finger Information (Obsolete)
- ☐ [0x100C] Dense Pack Searcher Information
- ☐ [0x100D] ETAK Information (Obsolete)
- ☐ [0x100E] Markov Statistics
- ☐ [0x100F] Searcher Information 2 (Obsolete)
- ☐ [0x1010] Handset Status (Obsolete)
- ☐ [0x1011] Vocoder Error Mask
- ☐ [0x1012] Analog Voice Channel
- ☐ [0x1013] Access Probe Information
- ☐ [0x1015] Test Mobile Command
- ☐ [0x1016] Sparse AGC Power Control
- ☐ [0x1017] Band Change
- ☐ [0x1019] General TA Information
- ☐ [0x101A] General TA with SCH Information
- ☐ [0x101C] Analog Forward Channel
- ☐ [0x101D] Analog Reverse Channel
- ☐ [0x101E] Analog Handoff Entry
- ☐ [0x101F] FM Slot Statistics Entry

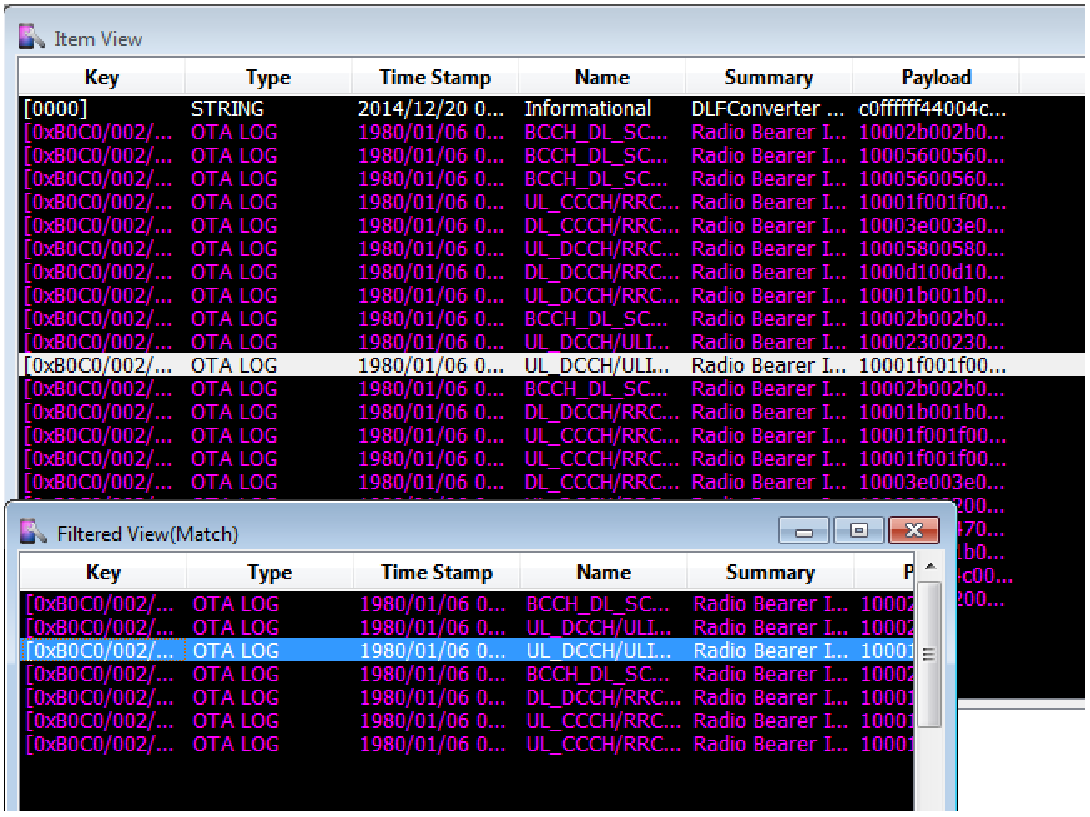
- **Raw Item** – This option displays the raw item contents in hexadecimal form for the currently selected item. A context menu can also be used to control the layout by clicking the right mouse button over the Raw Item Contents view.



```
00 47 00 C0 B0 00 38 7B 97 01 00 00 00   ..G.G.À°.8{—....
01 F2 01 2C 01 00 00 06 2E 00 22 12 15   ..P.ò.,......"..
00 FA 3A F0 0C 14 21 BA 8B 80 60 1A 22   ...ú:ð..!º‹`."
F9 9C 03 27 00 BE A2 67 03 2B F6 9A 1D   08.ùœ.'.¾¢g.+öš.
B3 81 95 FB 4D 2E 01 E0                  .ßa³ûM..à
```

- **Sync Near Item** – Sync Near Item tries to find the item with the given index in all item lists. In each list, if the item with the given index is found, it is selected. If the item with the given index is not found, the item with the index closest to the specified index is selected.

  It should be noted that item position correlates to generic timestamp, i.e., the item at index N has a generic timestamp that is earlier than the item at index N + 1, etc. In general, there is no correlation between item position and item-specific timestamps as they originate external to QXDM Pro.



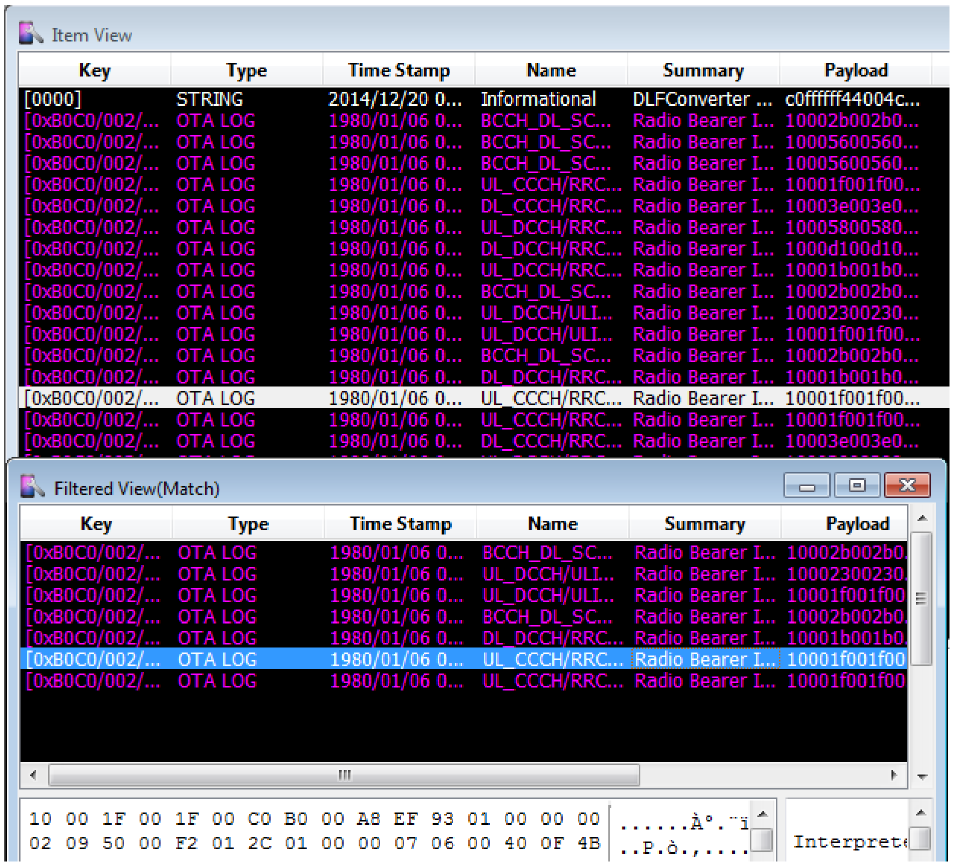
Item View

Key	Type	Time Stamp	Name	Summary	Payload
[0000]	STRING	2014/12/20 0...	Informational	DLFConverter ...	c0ffffff44004c...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002b002b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10005600560...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10005600560...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_CCCH/RRC...	Radio Bearer I...	10003e003e0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/RRC...	Radio Bearer I...	10005800580...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_DCCH/RRC...	Radio Bearer I...	1000d100d10...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/RRC...	Radio Bearer I...	10001b001b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002b002b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/ULI...	Radio Bearer I...	10002300230...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/ULI...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002b002b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_DCCH/RRC...	Radio Bearer I...	10001b001b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_CCCH/RRC...	Radio Bearer I...	10003e003e0...
					200...
					470...
					1b0...
					c00...
					200...

Filtered View(Match)

Key	Type	Time Stamp	Name	Summary	P
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/ULI...	Radio Bearer I...	10002
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/ULI...	Radio Bearer I...	10001
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_DCCH/RRC...	Radio Bearer I...	10001
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001

- **Sync To Item** – Sync all filtered views to the same selected item, if found. Sync To Item tries to find the item with the given index in all item lists. In each list, if the item with the given index is found, it is selected. If the item with the given index is not found, no change is made.



Item View

Key	Type	Time Stamp	Name	Summary	Payload
[0000]	STRING	2014/12/20 0...	Informational	DLFConverter ...	c0ffffff44004c...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002b002b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10005600560...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10005600560...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_CCCH/RRC...	Radio Bearer I...	10003e003e0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/RRC...	Radio Bearer I...	10005800580...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_DCCH/RRC...	Radio Bearer I...	1000d100d10...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/RRC...	Radio Bearer I...	10001b001b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002b002b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/ULI...	Radio Bearer I...	10002300230...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/ULI...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002b002b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_DCCH/RRC...	Radio Bearer I...	10001b001b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_CCCH/RRC...	Radio Bearer I...	10003e003e0...

Filtered View(Match)

Key	Type	Time Stamp	Name	Summary	Payload
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002b002b0
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/ULI...	Radio Bearer I...	10002300230
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/ULI...	Radio Bearer I...	10001f001f00
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002b002b0
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_DCCH/RRC...	Radio Bearer I...	10001b001b0
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001f001f00
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001f001f00

```
10 00 1F 00 1F 00 C0 B0 00 A8 EF 93 01 00 00 00   ......À°.¨ï
02 09 50 00 F2 01 2C 01 00 00 07 06 00 40 0F 4B   ..P.ò.,....
```

Interprete

- **Bookmark Item –** Mark selected items and adds them to the bookmark list view.

  To add a bookmark:

  a. Click **File→Open** and locate the HDF.

  b. Click **View→Common→Item View** (**F11**).

  c. Select an item to bookmark and press **ALT+B** . Alternatively, right-click the item and select **Bookmark Item.**

  To navigate using bookmarks:

  d. Click **File→Open** and select an HDF that has bookmarks.

  e. Click **View→Common →Item View** (**F11**).

  f. Press **ALT+U** to navigate up the bookmark list or **ALT+D** to navigate down the list.

NOTE: Navigation commands have no effect if the location of a bookmark is already being viewed and there are no additional bookmarks.

3.2.2.2 Raw Item pane

The Item View and Filtered View contain the Raw Item pane, which displays the raw item contents in hexadecimal form for the item currently selected in the Scrolling List pane.

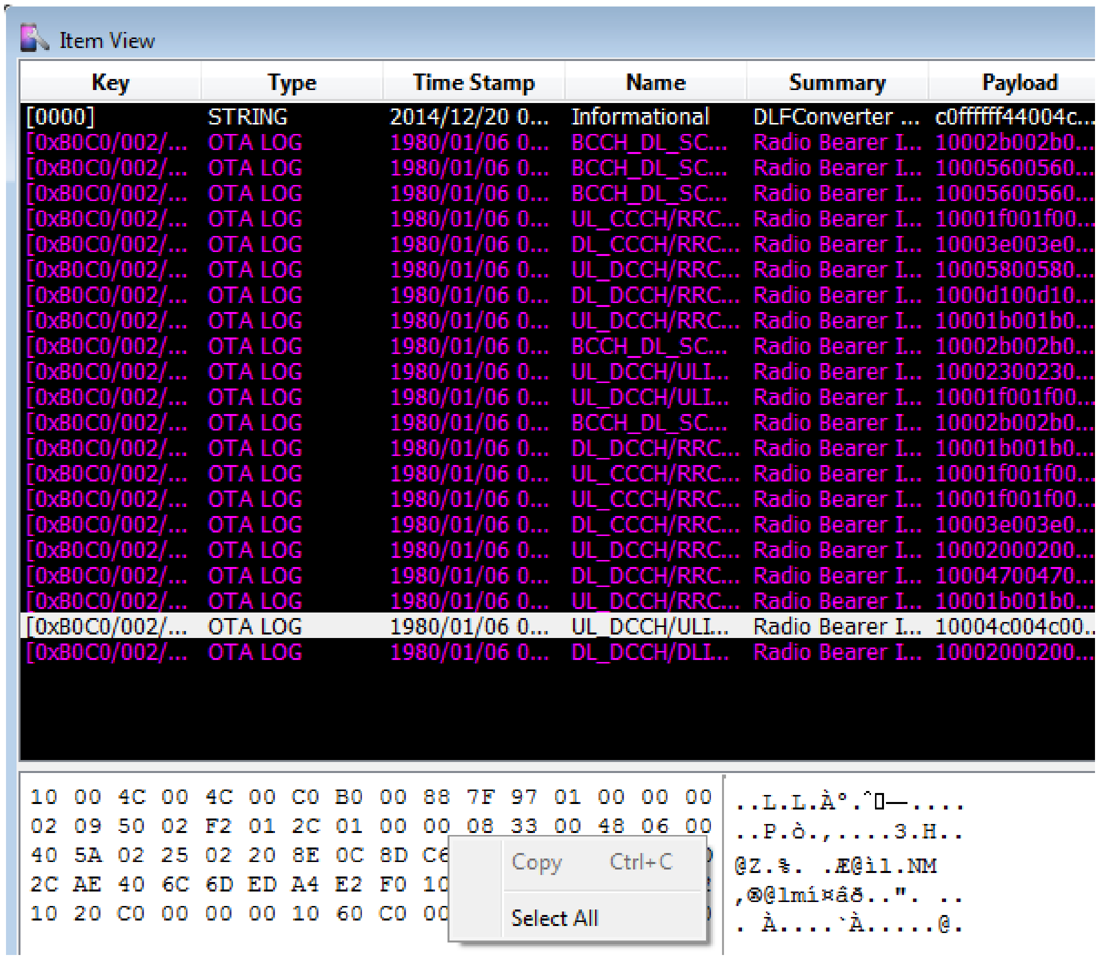
3.2.2.3 Parsed Item pane

The Item View and Filtered Views contain the Parsed Item pane. This pane displays the parsed fields or, when available, the parsed text of the currently selected item in the scrolling list pane.

Not all items have associated parsers to display parsed text in the parsed item pane. Item data is parsed using definitions in the QXDM Pro database, SILK, and ASN.1 for OTA log items, or (see Section 3.1).

Menu options

Right-clicking within the Parsed Item pane opens the context-sensitive menu.

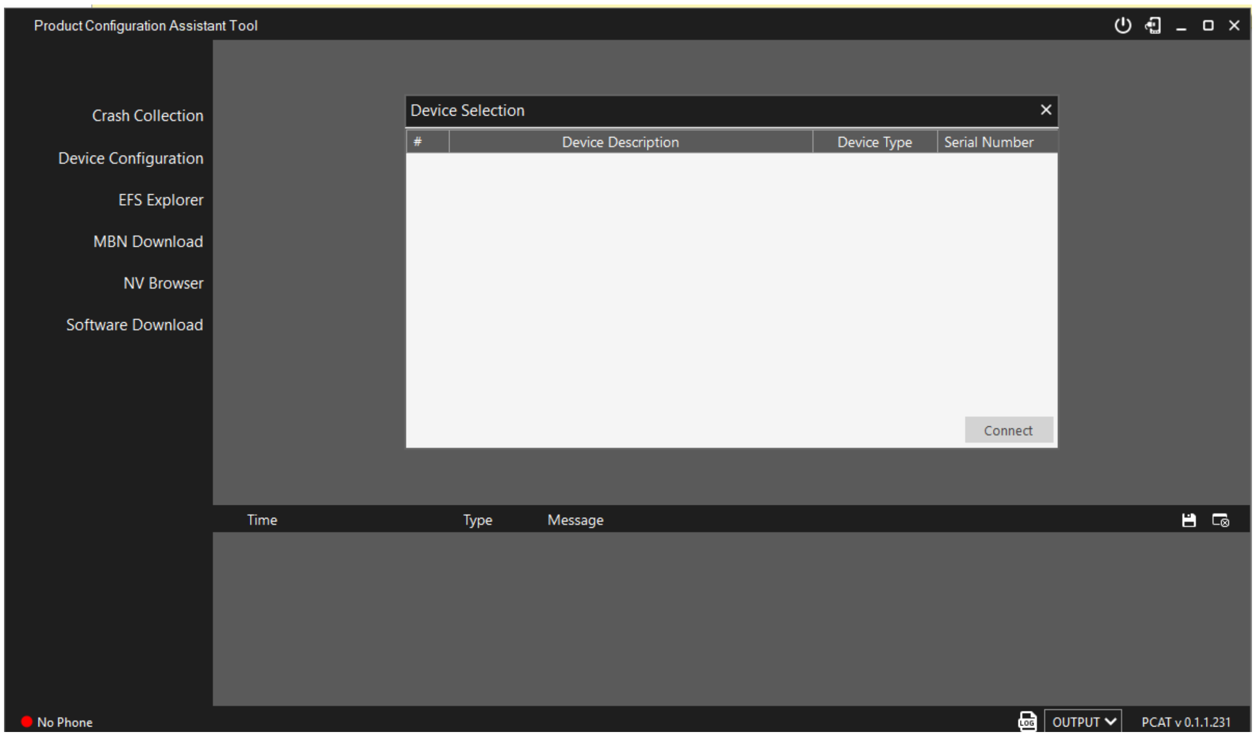


Key	Type	Time Stamp	Name	Summary	Payload
[0000]	STRING	2014/12/20 0...	Informational	DLFConverter ...	c0ffffff44004c...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002b002b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10005600560...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10005600560...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_CCCH/RRC...	Radio Bearer I...	10003e003e0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/RRC...	Radio Bearer I...	10005800580...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_DCCH/RRC...	Radio Bearer I...	1000d100d10...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/RRC...	Radio Bearer I...	10001b001b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002b002b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/ULI...	Radio Bearer I...	10002300230...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/ULI...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	BCCH_DL_SC...	Radio Bearer I...	10002b002b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_DCCH/RRC...	Radio Bearer I...	10001b001b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_CCCH/RRC...	Radio Bearer I...	10001f001f00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_CCCH/RRC...	Radio Bearer I...	10003e003e0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/RRC...	Radio Bearer I...	10002000200...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_DCCH/RRC...	Radio Bearer I...	10004700470...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/RRC...	Radio Bearer I...	10001b001b0...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	UL_DCCH/ULI...	Radio Bearer I...	10004c004c00...
[0xB0C0/002/...	OTA LOG	1980/01/06 0...	DL_DCCH/DLI...	Radio Bearer I...	10002000200...

```
10 00 4C 00 4C 00 C0 B0 00 88 7F 97 01 00 00 00   ..L.L.À°.ˆ▯—....
02 09 50 02 F2 01 2C 01 00 00 08 33 00 48 06 00   ..P.ò.,....3.H..
40 5A 02 25 02 20 8E 0C 8D C6                     @Z.%. .Æ@ìl.NM
2C AE 40 6C 6D ED A4 E2 F0 10                     ,®@lmí¤âð..". ..
10 20 C0 00 00 00 10 60 C0 00                     . À....`À.....@.
```

3.2.3 NV Browser

NV items stored in the nonvolatile memory of the connected target can be viewed and modified using PCAT (Product Configuration Assistant Tool), which is launched when clicking the **NV Browser** button.



Select a device from “Device Selection” window and click connect. PCAT Has NV Browser tab on the left to perform NV read/write operations.

To read an NV item, select the item to display the names of all the fields for that item and click the **Read** button to read the values from the phone. To write an NV item, select the item to display the names of all the item fields. To change the values, click the value in the Input column. After modifying the values, click the **Write** button to write the updated values back to the phone.

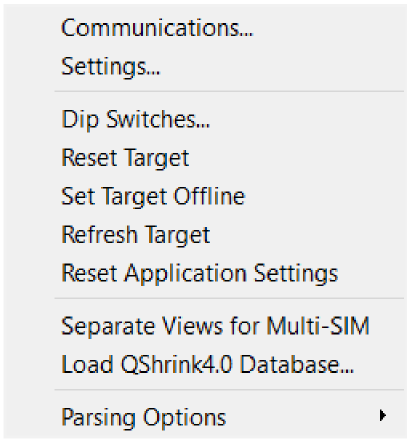
NV items can be sorted by clicking the column header. Columns can be reordered by holding down the left mouse over a column header and dragging it to a new location. The left-most column can be searched by typing the name or number of interest.

NOTE: Search typing is case-sensitive. The status of the NV read and write is given in the bottom left portion of the display. If clicking **Read** or **Write** yields DIAG Error Received or NV Status Error Received, then the target was not able to handle the request.

- **Offline** – Use the Offline button to send the Mode Change Request Offline command to the phone before writing NV items that have mode-specific requirements.
- **Reset** – Use the Reset button to send the Mode Change Request Reset command to the phone. This command when sent to a phone in Offline mode will result in the phone being recycled.
- **Read** – Use the Read button to read the selected NV item from the NV memory of the connected target.

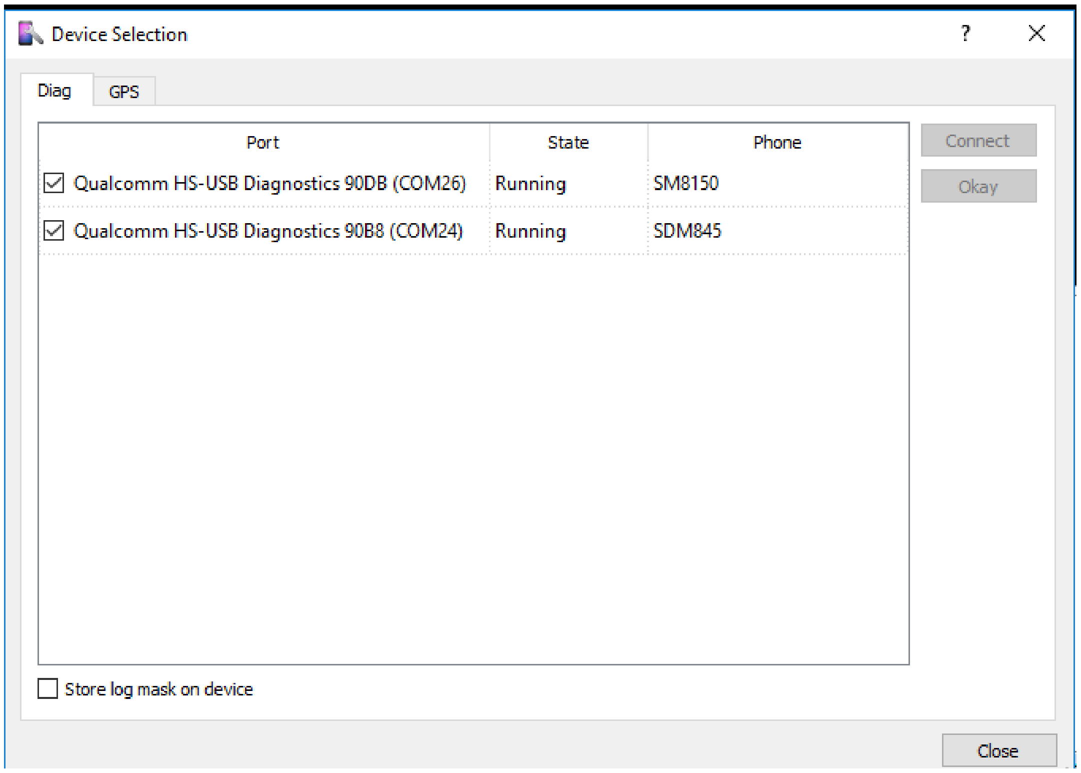
- **Write** – Use the Write button to write the selected NV item to the NV memory of the connected target.

3.3 Options menu



3.3.1 Communications

The Communications dialog, shown in the figure below, lists all ports in the system and the port properties, and allows users to choose the COM ports used by QXDM Pro for connection purposes.



Port	State	Phone
☑ Qualcomm HS-USB Diagnostics 90DB (COM26)	Running	SM8150
☑ Qualcomm HS-USB Diagnostics 90B8 (COM24)	Running	SDM845

☐ Store log mask on device

A table lists all ports in the system and their properties, including Port, State, and Phone.

- Store log mask on device

  When enabled, a .Diagin file will be generated in device to store the log masks used in QXDM session. This file is generally used to re-enable log masks when reset without connecting to QXDM.

3.3.2 Settings

Please refer to 3.1.10 - Item Store Settings

3.3.3 Dip Switch

Send the Dip Switch diagnostic command to device

3.3.4 Reset Target

Select the **Reset Target** option to send the mode change request reset command to the phone. This command, when sent to a phone in offline mode, results in the phone being recycled.

3.3.5 Set Target Offline

Send mode offline diagnostic command to device

3.3.6 Refresh Target

Re-establish QXDM session to device, as if new connection was made to device.

3.3.7 Reset Application Settings

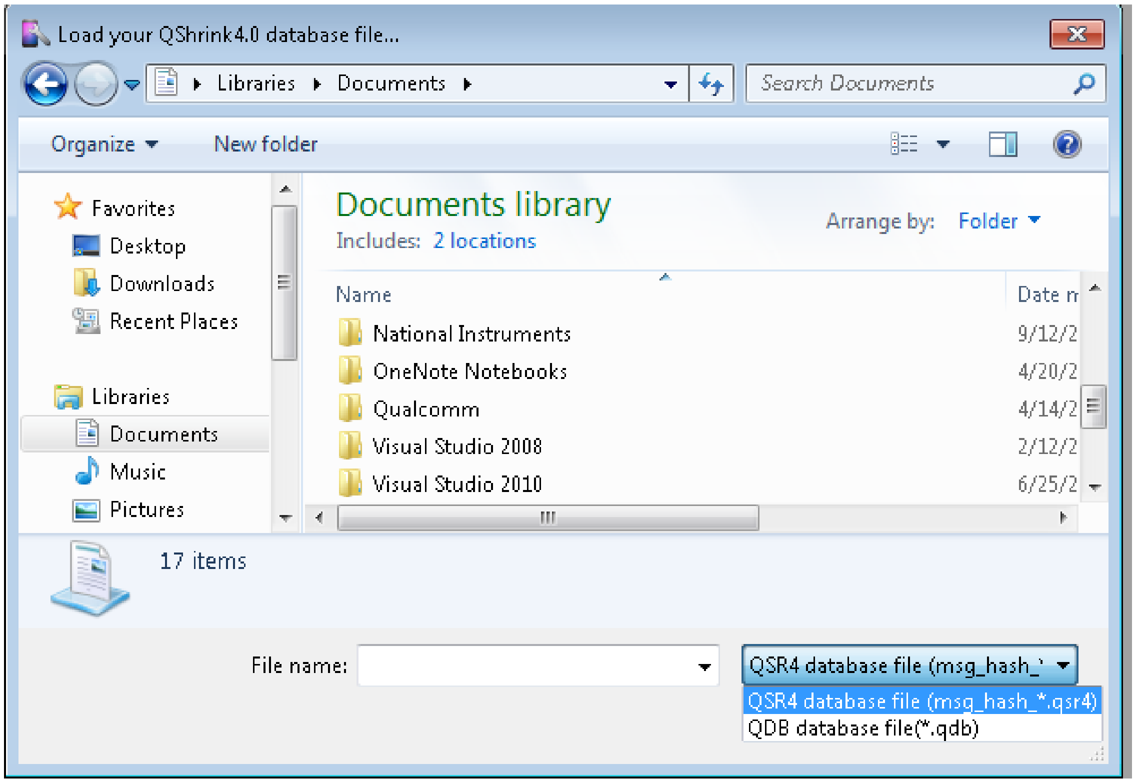
To reset QXDM tool setting to factory default

Separate Views for Multi-SIMEnabling this option triggers applicable views to be duplicated per SIM as new packets arrive. Disabling this option keeps all SIM information within a single view.

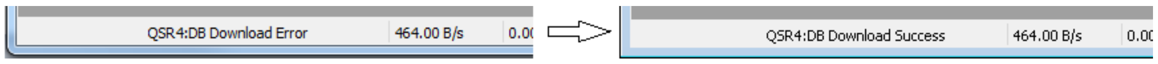
3.3.8 Load QShrink4 database

In the case where a specific QShrink4 database is to be used. User can manually load the QShrink4 database files into QXDM Pro. It supports two file types:

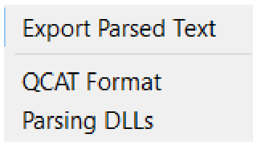
- Uncompressed file type *.qsr4
- Compressed file type *.qdb



If QXDM Pro is connecting to the device and the correct database files have been loaded, the QXDM Pro status bar will change the QShrink 4.0 database downloading status from “SR4: DB Download Error” to “QSR4: Download Success.” The QShrink4 debug messages can be correctly decoded after loading. The corresponding QShrink4.0 database files are copied to the /MessageHash folder.



3.3.9 Parsing option



3.3.9.1 Export Parsed Text

If enabled, when an item is copied, fully path, including header info will be copied.

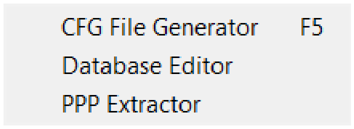
3.3.9.2 QCAT Format

If enabled, the compatible item content will be displayed in QCAT format.

3.3.9.3 Parsing DLLs

When selected, compatible item content will be displayed corresponding to DLL format. Location of the parsing DLL is in c:\ProgramData\Qualcomm\QXDM\Parsers

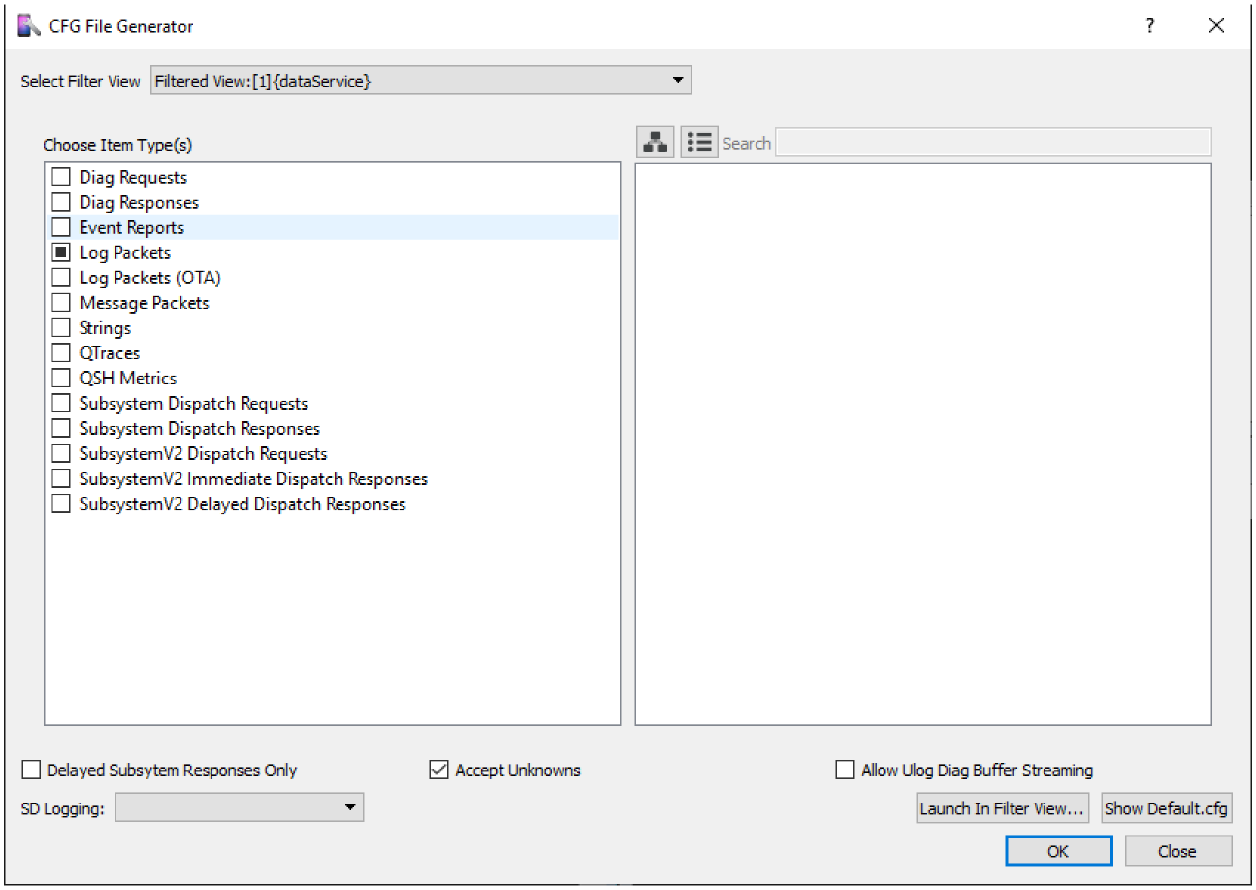
3.4 Tools menu



CFG File Generator F5
Database Editor
PPP Extractor

3.4.1 CFG File Generator

Function is used for generating an on-device log file.



Select Filter View: Filtered View:[1]{dataService}

Choose Item Type(s)

- [ ] Diag Requests
- [ ] Diag Responses
- [ ] Event Reports
- [■] Log Packets
- [ ] Log Packets (OTA)
- [ ] Message Packets
- [ ] Strings
- [ ] QTraces
- [ ] QSH Metrics
- [ ] Subsystem Dispatch Requests
- [ ] Subsystem Dispatch Responses
- [ ] SubsystemV2 Dispatch Requests
- [ ] SubsystemV2 Immediate Dispatch Responses
- [ ] SubsystemV2 Delayed Dispatch Responses

- [ ] Delayed Subsytem Responses Only
- [✓] Accept Unknowns
- [ ] Allow Ulog Diag Buffer Streaming

SD Logging:

3.4.1.1 Delayed Subsystem Responses Only

When selected, only the delayed subsystem response will be shown

3.4.1.2 SD Logging:

Save and load cfg file to a specific directory

3.4.1.3 Accept Unknowns

Device has logs that is not yet defined in QXDM database.

3.4.1.4 Allow Ulog Diag Buffer Streaming

Checking the box will unlock all Ulog Diag buffers for streaming, allowing them to work with ODL SD logging. It's the equivalent of checking all of the boxes in the "*View* -> *Common* -> *ULog* -> *Diag-ULog buffers config*" menu). This is to support the configuration in a .cfg file for SD logging, whereas the "*Diag-ULog buffers config*" is for dynamic interaction with user.

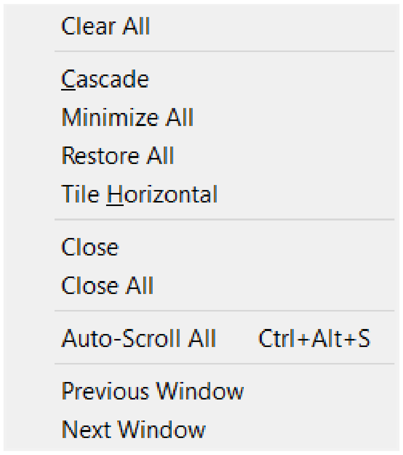
3.4.2 Database Editor

Launch DiagDbEditor application.

3.4.3 PPP Extractor

Launch PPP extractor application.

3.5 Window menu



3.5.1 Clear All

Data from applicable windows are cleared

3.5.2 Cascade

Windows that are opened are rearranged in a cascading order

3.5.3 Minimize All

All open windows are minimized

3.5.4 Restore All

All minimized windows are restored to their open settings

3.5.5 Tile Horizontal

Windows that are opened are rearranged in a horizontal fashion

3.5.6 Close

The window in focus are closed

3.5.7 Close All

All windows are closed

3.5.8 Auto Scroll All

Auto scroll all item, filtered, log, and message views.

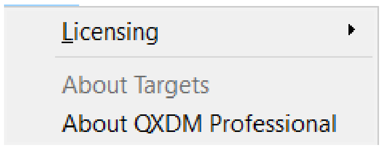
3.5.9 Previous Window

Focus is set to the previous logical window

3.5.10 Next Window

Focus is set to the next logical window

3.6 Help menu



3.6.1 Licensing

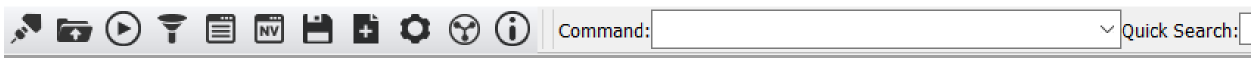
Licensing allows visibility of the status of the QLMS license. If the QXDM Pro QLMS license must be temporarily released on the current computer, select **Deactivate**.

NOTE: QLMS license deactivation requires an Internet connection. Attempting to deactivate the QLMS license without an internet connection will remove the license from the current computer but will still remain in the QLMS system. In this case, the only way to reclaim the license for use on another computer is to contact QLMS support and have the license manually removed from the QLMS system.

3.6.2 About QXDM Professional

Displays the version of QXDM tool

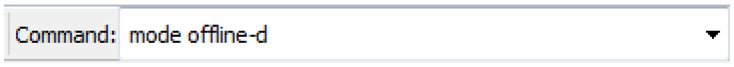
4 Icons & Indicators



- Communication – See section 3.3.1
- Open – See section 3.1.6
- Reply HDF – See section 3.1.9
- Filtered View – See section 3.2.2
- Item View – See section 3.2.2 3.2.1
- NV Browser – See section 3.2.3
- Save HDF – See section 3.1.8
- New Item – See section 3.1.7
- Create CFG File – See section 3.4.1 3.4.1
- Reset to Default DMC – See section 3.1.5
- Device Build Info
  - Display mobile build information. Info can also be found in View➔Common➔Device Information

4.1 Command

The Command bar provides access to the legacy QXDM Pro command interface.



Command: mode offline-d

Command interface

Although many legacy DOS DM script commands are supported through the QXDM Pro command interface. The recommended approach for automated testing is to use the automation interface (see Chapter 5).

Script commands can be typed in to the Command edit control. The Command Output display and the Item View display script output. Additionally, any Filtered View configured for the item type "Strings" and the item key "Automation" displays script output.

Script Help for command interface

The Script Help page documents all supported commands. It is available in the View combo-box from the View bar. This view also provides documentation for the legacy script commands that are part of the original DOS DM scripting interface.

NOTE: QXDM Pro property definitions are not sufficient to describe many diagnostic packets and have been migrated to the QXDM Pro database. User-defined property definitions will also need to be migrated to user database definitions

NOTE: Table 4-1 lists the replacements that should be used instead of the obsolete property interface commands.

Table 4-1 Property command replacements

QXDM3	QXDM4
RequestItemFile RequestNamedItem RequestNamedItemFile	

4.1.1 Command functions

The Script Help page documents all supported command prompt functions. It is available in the View combo box from the View Bar. This view also provides documentation for the legacy script commands that are part of the original DOS DM scripting interface.

Some script commands direct packets to the test phone, while others affect only QXDM Pro operations. Command output is displayed in the Command Output window and can also be viewed from the Item View and any Filtered View that has registered interest in automation string items.

Several functions are provided to support waiting for, requesting, and parsing of any item that is described in the QXDM Pro database. Additional support is provided in the Item Tester application that provides formatting examples using these commands for all items described in the database. The Item Tester is accessible from the QXDM Pro Tools menu.

4.1.1.1 CWait

This halts scripting for a specified amount of time.

Parameter	Description
Time	Time in centiseconds to wait

Usage example:

```
Wait 500
Halt scripting for 5 sec
```

4.1.1.2 Echo

Adds a text string to the DM Item Store.

Parameter	Description
Text	Adds the provided string to the DM Item Store with the Automation type; the string will appear in a list-based view registered for this type (including the Command Output view)

Usage example:

```
Echo “Test completed”
```

4.1.1.3 Mode

This requests that the target perform a mode change.

Parameter	Description
Mode	Mode can be one of the following string values: ▪ FTM – Change to Factory Test mode ▪ LPM – Change to Low Power mode ▪ Offline-A – Change to Offline Analog mode ▪ Offline-D – Change to Offline Digital mode ▪ Online – Change to Online Digital mode ▪ Reset – Reset the target (may require Phone Offline mode)

Usage example:

Mode Reset – Requests the target reset

4.1.1.4 Pause

This halts processing of a legacy script (.scr) processing. The Space Bar must be pressed in the Command edit box to continue processing.

Usage example:

```
Pause
```

4.1.1.5 RequestItem

RequestItem allows the scheduling of a request to be sent to the phone. The request itself is built according to a description entered into the QXDM Pro database or the user database diag entity table. RequestItem waits for a response and, if received, parses it according to the default description in the QXDM Pro database or the user database (if a description exists).

Parameter	Description
Name	Name of diag entity to request (entity must be either a diag request or subsystem dispatch request)
Fields	List of space-separated fields used to build the request

Usage example:

```
RequestItem “Version Number Request”
RequestItem “Extensible Parameter Retrieval Request” 1 1 100
```

4.1.1.6 RequestNVItemRead

RequestNVItemRead allows for one NV read request to be scheduled to be sent to the phone. The request itself is built according to a description entered into the QXDM Pro database or the user database diag entity table. RequestNVItemRead waits for the response and, if received, parses it according to the description in the QXDM Pro database or the user database diag entity tables.

Parameter	Description
NVItemName	Name of the NV item entity to read
Fields	Field values used to build the request

Usage example:

```
RequestNVItemRead “banner”
RequestNVItemRead “air_timer” 1
```

4.1.1.7 RequestNVItemWrite

RequestNVItemWrite allows for one NV write request to be scheduled to be sent to the phone. The request itself is built according to a description entered into the QXDM Pro database or the user database diag entity table. RequestNVItemWrite waits for the response and, if received, parses it according to the description in the QXDM Pro database or the user database diag entity tables.

Parameter	Description
NVItemName	Name of the NV item entity to write
Fields	Field values used to build the request

Usage example:

```
RequestNVItemWrite “banner” “My MSM7500”
```

4.1.1.8 RequestNVItemIDRead

RequestNVItemIDRead allows one to schedule one NV read request to be sent to the phone. The request itself is built according to arguments entered as raw bytes. RequestNVItemIDRead waits for the response, reporting result status.

Parameter	Description
ItemNumber	Item number of the NV diag entity to be read
RawBytes	Optional list of space-separated byte values used to build the NV item being read

Usage example:

```
RequestNVItemIDRead 71
RequestNVItemIDRead 10 0 0 0
```

4.1.1.9 RequestNVItemIDWrite

RequestNVItemIDWrite allows one to schedule one NV write request to be sent to the phone. The request itself is built according to arguments entered as raw bytes. RequestNVItemIDWrite waits for the response, reporting result status.

Parameter	Description
ItemNumber	Item number of the NV diag entity to be written
RawBytes	Optional list of space-separated byte values used to build the NV item being written

Usage example:

```
RequestNVItemIDWrite 71 77 83 77 32 54 53 53 48
RequestNVItemIDWrite 10 0 0 0
```

4.1.1.10 SendRawRequest

SendRawRequest allows one to schedule one diag request to be sent to the phone. The request itself is built according to arguments entered as raw bytes. SendRawRequest waits for the response, reporting result status.

Parameter	Description
RawBytes	List of space-separated byte values used to build the diag request being sent

Usage example:

```
SendRawRequest 39 71 0 77 83 77 32 54 53 53 48
SendRawRequest 0
```

4.1.1.11 Wait

This halts scripting for a specified amount of time.

Parameter	Description
Time	Time in seconds to wait

Usage example:

```
Wait 10
```

4.1.1.12 WaitForItem

WaitForItem allows you to register interest in and wait for an item. The item name is the diag entity name in the QXDM Pro database or the user database diag entity table. Requests for asynchronous commands (logs, events, and debug messages) may result in a registration being sent to the phone. WaitForItem can also be used to wait for synchronous requests or responses. At the conclusion of the command, the registration is revoked.

NOTE: WaitForItem only waits for a response. It does not make a request. Use RequestItem (see Section 4.1.1.5) or RequestNamedItem to request an item and wait for the response.

Parameter	Description
ItemName	Name of diag entity to request

Usage example:

```
WaitForItem EVENT_CALL_CONTROL_TERMINATED
WaitForItem "Searcher and Finger"
```

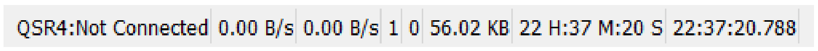
WaitForItem “Get IS-2000 Status Response”

4.2 Quick Search

Enable Quick search on views and view launch

4.3 Status bar

The Status bar displays a configurable subset of application statistics, application state indicators, and help for select menu commands.



QSR4:Not Connected | 0.00 B/s | 0.00 B/s | 1 | 0 | 56.02 KB | 22 H:37 M:20 S | 22:37:20.788

Status from left to right

- Qshrink4 usage status
- RX rate
- TX rate
- Total number of packet received
- Total number of packet dropped by device

- Total size of HDF file
- Duration of the current QXDM session
- Time stamp of the last received packet.

5 COM Automation Interfaces

In QXDM5, automation should be done via QUTS APIs, except a few legacy API from QXDM related to UI operation. For sample automation scripts, please refer to C:\Program Files (x86)\Qualcomm\QUTS\SampleCode.

The QXDM Pro application is a Windows Vista (and later) application that utilizes the existing diag serial interface via the QUTS phone server to establish a connection to a phone target. Once a connection is established, QXDM Pro allows the user to manipulate and monitor the phone target by ultimately sending and receiving diag traffic to and from the target. QXDM Pro is built upon the DM Core Framework, a library of objects and methods that encompasses all nonuser interface-related functionality implemented by QXDM Pro. This includes but is not limited to target connection management, target data partitioning, target data representation, and data storage and nonsequential data access.

To support extensibility and rapid prototyping, QXDM Pro exports a subset of the DM Core Framework functionality through a set of COM interfaces. These interfaces allow QXDM Pro to support scripting through any COM-compliant scripting language, e.g., VBScript, JScript, and Perl. Additionally, a large subset of these interfaces is exposed through HTML-based QXDM Pro views. Each view combines the capabilities of the Microsoft Internet Explorer control, hosted in a QXDM Pro window, with the rich QXDM Pro COM interface capabilities, allowing both the QXDM Pro team and QXDM Pro users to create custom views.

The QXDM Pro COM interfaces are roughly divided into five modules, each described in the following sections:

- **QXDM Professional** – The QXDM Pro COM automation interface that existed and shipped in QXDM versions prior to 03.05.50; although these interfaces are still part of the QXDM code base, it is intended that support for some of the interfaces will be phased out over time
- **Client** – Provides a means to partition QXDM Pro items (data obtained from the target and internal QXDM Pro processing) into unique collections; also provides direct access to QXDM Pro items
- **Client Config** – Provides a means to configure the particular types of items in which a QXDM interface client is interested; i.e., it defines the partition a client represents
- **Item** – Interface to a particular item being managed by QXDM Pro
- **Field** – Provides access to the QXDM Pro database parsed fields of a particular item

NOTE: Numerous automation sample scripts (written in both Perl and JScript) can be found under the executable folder described in Section 2.2 (which is typically located in C:\Program Files (x86)\Qualcomm\QXDM4\Automation Samples).

5.1 QXDM Pro interface

This section describes the legacy QXDM Pro COM interface.

5.1.1 AppVersion

The AppVersion property returns the version of QXDM Pro.

A BSTR value is returned.

5.1.2 ClearViewItems

The ClearViewItems() function clears all items in the specified view.

Parameter	Type	Description
Display Name	BSTR	Name of display to clear; the name is interpreted as: ▪ Item view – Item Store view ▪ Messages view – Messages view ▪ All others – Tagged filtered view name

The function returns a VARIANT_BOOL response:

- FALSE – Request was not received (e.g., view was not found)
- TRUE – Request was successfully received

5.1.3 CloseView

The CloseView() function allows you to close an existing QXDM Pro view.

Parameter	Type	Description
pViewName	BSTR	Name of QXDM Pro view to close
pViewTag	BSTR	Window tag for views that support multiple instances or empty string (“”)

The function returns a VARIANT_BOOL that indicates whether a view was closed.

NOTE: Due to the unique operation of the QXDM Pro log view, this function does not apply to the QXDM Pro log view.

5.1.4 CreateView

The CreateView() function allows you to bring up a QXDM Pro view.

Parameter	Type	Description
pViewName	BSTR	Name of QXDM view to bring up
pViewTag	BSTR	Window tag for views that support multiple instances or empty string (““)

The function returns a VARIANT_BOOL that indicates whether a view was created. If a view of the same name is already present and that view does not support multiple instances, the existing view will be brought to the forefront. In this case, success is returned.

NOTE: Due to the unique operation of the QXD QXDM Pro M log view, this function does not apply to the QXDM Pro log view.

5.1.5 setVisible()

The setVisible() takes a Boolean as parameter. If parameter is true, the function will set the current QXDM window visible. Else current QXDM window will be invisible, but simply running in the background. User can see the instance running in Task Manager in both scenarios.

Parameter	Type	Description
visible	Boolean	▪ if true turns current QXDM window visible, else invisible.

The functions does not have a return type.

5.1.6 getVisible()

The getVisible() returns if QXDM window is visible or just running in the background.

The function takes no parameters.

The functions return a Boolean, true if current QXDM window is visible, false otherwise.

5.1.7 ExportViewText

The ExportViewText() function exports all items to a text file.

Parameter	Type	Description
Display Name	BSTR	Name of display to export; the name is interpreted as: ▪ Item view – Item Store view ▪ Messages view – Messages view ▪ Log view – Log view ▪ All others – Tagged filtered view name
Destination File	BSTR	Name of the destination text file

The function returns a VARIANT_BOOL response:

- FALSE – Request was not received (e.g., view was not found)
- TRUE – Request was successfully received

5.1.8 CopyViewItems

The CopyViewItems() function copies all items to an Item Store Format file.

Parameter	Type	Description
Display Name	BSTR	Name of display to copy; the name is interpreted as: ▪ Item view – Item Store view ▪ Messages view – Messages view ▪ Log view – Log view ▪ All others – Tagged filtered view name
Destination File	BSTR	Name of the destination Item Store Format file

The function returns a VARIANT_BOOL response:

- FALSE – Request was not received (e.g., view was not found)
- TRUE – Request was successfully received

5.1.9 LoadConfig

The LoadConfig() function loads a configuration file (.DMC).

Parameter	Type	Description
Config File	BSTR	Configuration file (see Section 3.1 for details)

The function returns no response.

5.1.10 SaveConfig

The SaveConfig() function saves a configuration file (.dmc).

Parameter	Type	Description
Config File	BSTR	Configuration file (see Section 3.1 for details)

The function returns no response.

5.1.11 openFile

This API is equivalent to open a file using QXDM UI.

Parameter	Type	Description
filePath	String	Fully qualified path to file

The function has no return type.

5.1.12 LoadItemStore

The LoadItemStore() function allows you to load an Item Store Format (*.HDF) file into QXDM Pro.

Parameter	Type	Description
pFileName	BSTR	Fully qualified path to Item Store Format file

The function returns a VARIANT_BOOL that indicates whether the Item Store Format file was loaded.

NOTE: An HDF can be loaded into QXDM Pro only when QXDM Pro is in the disconnected state.

5.1.13 SaveItemStore

The SaveItemStore() function allows you to save a temporary item store format file (*.HDF). This function is subject to the same conditions as Save Items (see Figure 3-1).

Parameter	Type	Description
pFileName	BSTR	Fully qualified file name of the Item Store Format file to be saved

The function returns no value.

NOTE: When QXDM Pro is run, an HDF is created using a temporary name in the QXDM Pro HDF folder (typically located at C:\Documents and Settings\All Users\Documents\Qualcomm\QXDM\HDF). This is then used as the current HDF. If the user (either through the interaction with the QXDM Pro GUI or through the QXDM Pro automation interface) initiates the loading of an existing HDF, the current HDF may be saved if it is nonempty and the user has enabled Item Store saving. In this case, the filename specified by SaveItemStore() is used to rename and move the current HDF. This process does not apply when the user has already loaded an existing HDF.

5.1.14 QuitApplication

The QuitApplication() function causes the QXDM Pro application to terminate.

The function returns no value.

5.2 Device connection

This section describes the legacy QXDM Pro COM interface.

5.2.1 ConnectToDevice

The ConnectToDevice () function connects a device to QXDM

Parameter	Type	Description
deviceHandle	string	Device handle of the device to be connected
protHandle	String	Protocol handle of the device to be connected
bIsGpsDevice	Boolean	If this is a GPS device

This function returns true if device is connected successfully, else false.

5.2.2 disconnectFromDevice

The disconnectFromDevice() function disconnects the device to QXDM

This function does not take parameters.

This function has no return type.

5.2.3 GetIsPhoneConnected

The GetIsPhoneConnected() function returns true is a device is connected to QXDM, false otherwise.

This function takes no parameters.

5.3 Item Store advance options

This section describes the legacy QXDM Pro COM interface.

5.3.1 SetItemStoreAdvanceOptions

The SetItemStoreAdvanceOptions () function sets the properties for item store settings.

Parameter	Type	Description
enableISFAdvanceMode	boolean	set true to enable isf advanced mode, false otherwise
maxISFfileSizeInMBs	Unsigned int	isf file size in mb
maxISFDuration	Unsigned int	max isf file duration
autosaveISF	Boolean	set true to enable isf auto save, false otherwise
maxISFArchives	Unsigned int	max number of isf file archive

This function has no return type.

5.3.2 SetISFFileSize

The `SetISFFileSize` () function sets size of ISF when QXDM automatically save the file

Parameter	Type	Description
fileSizeInMBs	Unsigned int	Set size of ISF file that trigger saving the file

This function has no return type.

5.3.3 GetISFFileSize

The `GetISFFileSize` () function returns size of ISF when QXDM automatically save the file

This function returns an unsigned int of the size of the ISF in MB.

5.3.4 SetAutoSaveISF

The `SetAutoSaveISF` () function enable ot disable auto save ISF file.

Parameter	Type	Description
Autosave	Boolean	If true sets auto save to be true, else false.

This function has no return type

5.3.5 GetAutoSaveISF

The `GetAutoSaveISF` () function returns if auto saving of isf file is enabled

This function returns a Boolean to show if auto saving is enabled.

5.3.6 SetISFMaxDuration

The `SetISFMaxDuration` () function sets the duration that trigger auto saving of ISF

Parameter	Type	Description
maxDuration	Unsigned int	Number of minutes before auto saving is triggered

This function returns a Boolean to show if auto saving is enabled.

5.3.7 GetSFMaxDuration

The `GetISFMaxDuration` () function returns the number of minutes before auto saving is triggered.

5.3.8 SetMaxISFArchives

The `SetMaxISFArchives` () function sets the max number of files for auto save operation

Parameter	Type	Description
maxIsfArchives	Unsigned int	Number of files for auto save

This function has no return type

5.3.9 GetMaxISFArchives

The `GetMaxISFArchives` () function returns the max number of files for auto save operation

5.3.10 EnableISFAdvancedMode

The `enableISFAdvanceMode` () function enable or disable auto save operation

Parameter	Type	Description
enableIsfAdvanceMode	boolean	If advanced auto saving is enabled

This function has no return type

5.3.11 IsISFAdvancedModeEnabled

The `IsISFAdvancedModeEnabled` () function returns if advanced auto saving mode is enabled.

The function returns a Boolean, true if advanced mode is enabled, false otherwise.

5.3.12 SetParsingOrder

The `SetParsingOrder` () function sets the max number of files for auto save operation

Parameter	Type	Description
Order	Unsigned int	0: QXDM parsed option, 1. QCAT/APEX parsing, 2. Using parsing DLLs

This function returns 1 if parsing preference is set. 0 otherwise

5.3.13 SaveLogViewAfterClose

The `SaveLogViewAfterClose` () function enable or disable automatic log saving after log view is closed

Parameter	Type	Description
saveLogViewContent	Boolean	If true turns on automatic saving of log view packets; else will turn off

This function has no return type

5.3.14 SetBaseISFFileName

The `SetaseISFFileName` () function sets the prefix of the file name to be saved

Parameter	Type	Description
isfFileName	String	The prefix of the name for the file to be saved

This function has no return type

5.3.15 GetBaseISFFileName

The `GetaseISFFileName` () function returns the prefix of the file name to be saved

This function returns a string of the prefix for the file name.

5.3.16 SetISFDirPath

The `SetISFDirPath` () function sets the file path for the file to be saved

Parameter	Type	Description
isfDirPath	String	The full isf file path

This function has no return type

5.3.17 GetISFDirPath

The `GetISFDirPath` () function returns the file path for the file to be saved

This function returns a string representing the full file path for the file.

5.4 Item interface (IColorItem)

This section describes the QXDM Pro Item interface (IColorItem).

5.4.1 GetItemType

The GetItemType() function allows you to retrieve the item type of the current color item. The function takes no arguments and returns a ULONG value. The ULONG return value is one of the values shown in Table 5-1.

Table 5-1 GetItem Type return values

Value	Description
0	Malformed diag entity
1	Target diag response (minus following)
2	Target diag request (minus following)
3	GPS information
4	Target event
5	Target log
6	Target message
7	Generic strings
8	Target OTA log
9	Target subsystem dispatch response
10	Target subsystem dispatch request
0xFFFFFFFF	Error

5.4.2 GetItemTypeText

The GetItemTypeText() function allows you to retrieve the item type converted to a string. The function takes no arguments and returns a BSTR value.

The string returned will be identical to the contents of the Type column in any QXDM Pro list-based view.

5.4.3 GetItemColor

The GetItemColor() function allows you to retrieve the color of the current color item. The function takes no arguments and returns a ULONG value. The ULONG return value is in the form of a standard Windows COLORREF value, i.e., three 8-bit values (R, G, B) packed into a 32-bit space. Upon return, (0, 0, 0) is returned. This represents black, an invalid value in QXDM Pro, as that is the color of all item list backgrounds.

5.4.4 GetItemTimestamp

The GetItemTimestamp() function allows you to retrieve the generic timestamp of the current color item. The function takes no arguments and returns a standard Windows DATE value (VT_DATE).

The generic timestamp represents the time the color item was created, i.e., when it first entered the current Item Store. This is not the same as the item-specific timestamp, which is assigned to the item by the target.

5.4.5 GetItemTimestamp2

The GetItemTimestamp2() function allows you to retrieve the generic timestamp of the current color item. The function takes no arguments and returns a standard Windows DATE value as a double precision floating point number (VT_R8).

The generic timestamp represents the time the color item was created, i.e., when it first entered the current Item Store. This is not the same as the item-specific timestamp, which is assigned to the item by the target.

5.4.6 GetItemTimestampText

The GetItemTimestampText() function allows you to retrieve the generic item timestamp converted to a string. The function returns a BSTR.

Parameter	Type	Description
bWantDate	VARIANT_BOOL	Include the date
bWantMillisecs	VARIANT_BOOL	Include ms

The string returned will be identical to the contents of the Timestamp column in any QXDM Pro list-based view.

5.4.7 GetItemSpecificTimestamp

The GetItemSpecificTimestamp() function allows you to retrieve the item-specific timestamp of the current color item. The function takes no arguments and returns a standard Windows DATE value (VT_DATE).

The item-specific timestamp represents the time assigned to the item by the target. If the item does not support a target-assigned timestamp, then the generic timestamp will be returned instead.

5.4.8 GetItemSpecificTimestamp2

This function allows you to retrieve the item-specific timestamp of the current color item. The function takes no arguments and returns a standard Windows DATE value as a double precision floating point number (VT_R8).

The item-specific timestamp represents the time assigned to the item by the target. If the item does not support a target-assigned timestamp, then the generic timestamp will be returned instead.

5.4.9 GetItemSpecificTimestampText

The GetItemSpecificTimestampText() function allows you to retrieve the item-specific timestamp converted to a string.

Parameter	Type	Description
bWantDate	VARIANT_BOOL	Include the date
bWantMillisecs	VARIANT_BOOL	Include milliseconds

A BSTR value is returned.

The string returned will be identical to the contents of the Timestamp column in any QXDM Pro list-based view.

5.4.10 GetItemBuffer

The GetItemBuffer() function allows you to retrieve the raw buffer of the current color item.

Parameter	Type	Description
bIncludeHeaderBytes	VARIANT_BOOL	Return the full item buffer, including the header defined by QXDM Pro or simply the payload defined by QXDM Pro

The function returns a VARIANT array (VT_ARRAY | VT_UI1) when successful. When an error occurs, an empty VARIANT array is returned. However, as the payload defined by QXDM Pro may itself be empty, this is not a reliable way of determining if a true error occurred.

5.4.11 GetItemDLFBuffer

The GetItemDLFBuffer() function allows you to retrieve the raw buffer of the current color item adjusted to be compatible with the legacy DLF log file format.

The function returns a VARIANT array (VT_ARRAY | VT_UI1) when successful. When an error occurs, an empty VARIANT array is returned.

5.4.12 GetItemBufferText

The GetItemBufferText() function allows you to retrieve the raw buffer of the current color item converted to a string. The function returns a BSTR.

Parameter	Type	Description
bIncludeHeaderBytes	VARIANT_BOOL	Return the full item buffer, including the header defined by QXDM Pro or simply the payload defined by QXDM Pro.

The string returned will be identical to the contents of the Payload column in any QXDM Pro list-based view.

5.4.13 GetItemKeyText

The GetItemKeyText() function allows you to retrieve the item key converted to a string. The function takes no arguments and returns a BSTR value.

The string returned will be identical to the contents of the Key column in any QXDM Pro list-based view.

5.4.14 GetItemName

The GetItemName() function allows you to retrieve the item name. The function takes no arguments and returns a BSTR value.

The string returned will be identical to the contents of the Name column in any QXDM Pro list-based view. The item name represents the item key mapped to a string defined in the appropriate QXDM Pro database category/reference table.

5.4.15 GetItemSummary

The GetItemNameSummary() function allows you to retrieve the item summary. The function takes no arguments and returns a BSTR value.

The string returned will be identical to the contents of the Summary column in any QXDM Pro list-based view. The item summary represents a summary of the item payload generated either by internal QXDM Pro processes, the QXDM Pro database structure definition in combination with a QXDM Pro database format specifier, or an external dynamic item parsing DLL.

5.4.16 GetItemSize

The GetItemSize() function allows you to retrieve the item size of the current color item. The function returns a ULONG value representing the size in bytes of the item.

Parameter	Type	Description
bIncludeHeaderBytes	VARIANT_BOOL	Return the full item size, including the header defined by QXDM Pro or simply the payload defined by QXDM Pro sessional

5.4.17 GetItemParsedText

The GetItemParsedText() function allows you to retrieve the item full parsed text. The function takes no arguments and returns a BSTR value.

The string returned will be identical to the contents of the bottom right pane in any QXDM Pro list-based view (currently excludes the database parsed field list). The item full parsed text is a full description of the item payload generated either by internal QXDM Pro processes, the QXDM Pro database structure definition, an external dynamic item parsing DLL, or (in the case of OTA logs) SILK.

5.4.18 GetItemFieldValue

The GetItemFieldValue() function allows you to retrieve the actual value of a field parsed out of the item buffer. Typical applications of this function would be to parse items that cannot be described by the database or to extract fields out of the header of an item.

Parameter	Type	Description
fieldType	ULONG	Type of field to obtain (see Table 8-8)
bitCount	ULONG	Number of bits to extract
bitOffset	ULONG	Offset into item buffer to begin extraction
bIncludeHeaderBytes	VARIANT_BOOL	Include the header in item buffer

The function returns a VARIANT. In the case of an error, an empty VARIANT (VT_EMPTY) is returned. Table 5-2 shows the types of supported fields.

Table 5-2 Supported field types

Type	Value	VARIANT type	Description
BOOL	0	VT_BOOL	Boolean (true/false)
INT8	1	VT_I1	8-bit signed value
UINT8	2	VT_UI1	8-bit unsigned value
INT16	3	VT_I2	16-bit signed value
UINT16	4	VT_UI2	16-bit unsigned value
INT32	5	VT_I4	32-bit signed value
UINT32	6	VT_UI4	32-bit unsigned value
INT64	7	VT_I8	64-bit signed value
UINT64	8	VT_UI8	64-bit unsigned value
ANSI String	9	VT_BSTR	ANSI string (fixed length given by bitCount)
UNICODE String	10	VT_BSTR	UNICODE string (fixed length given by bitCount)
ANSI String	11	VT_BSTR	ANSI string (NULL terminated, bitCount ignored)
UNICODE String	12	VT_BSTR	UNICODE string (NULL terminated, bitCount ignored)
FLOAT32	13	VT_R4	IEEE 32-bit floating point value
FLOAT64	14	VT_R8	IEEE 64-bit floating point value

5.4.19 GetItemFieldValueText

The GetItemFieldValueText() function allows you to retrieve the text representation of the field value parsed out of the item buffer.

Parameter	Type	Description
fieldType	ULONG	Type of field to obtain (see Table 5-2)
bitCount	ULONG	Number of bits to extract
bitOffset	ULONG	Offset into item buffer to begin extraction
bIncludeHeaderBytes	VARIANT_BOOL	Include the header in item buffer
bHexadecimal	VARIANT_BOOL	Format field text as hexadecimal

The function returns the field value as a VT_BSTR and operates in the same manner as GetItemFieldValue described above.

5.4.20 GetItemFields

The GetItemFields() function allows you to retrieve the QXDM Pro structure database representation of the current color item. The function takes no arguments and returns an interface pointer to the QXDM Pro database parsed field interface model (IColorItemFields) as discussed in Section **Error! Reference source not found.**.

5.4.21 GetConfiguredItemFields

The GetConfiguredItemFields() function allows you to retrieve the database parsed item fields of the current color item. For applications that process a large number of items per sec or process items that parse to a large number of fields, setting bFieldStrings and/or bFieldNames to false will improve performance and/or is usually dependent on whether you need field strings or not. Generally, you only need field strings if you are dealing with a structure that contains enumerations and you need to display the mapped enumeration value strings.

Parameter	Type	Description
pEntityName	LPCTSTR	Diag entity name (if this is not given then the default structure associated with this entity will be used, i.e., GetItemFields() behavior)
bFieldStrings	VARIANT_BOOL	Generate string representations of field values
bFieldNames	VARIANT_BOOL	Generate partial field names

The function returns an interface pointer to the QXDM Pro database parsed field interface model, IColorItemFields (Section **Error! Reference source not found.**).

NOTE: If bFieldStrings is set to false, retrieving the string representation of fields will not be available. If bFieldNames is set to false, searching for the index of a field by name will not be available.

A References

A.1 Acronyms and terms

Acronym or term	Definition
AMSS	Advanced mobile subscriber station
APU	Application processing unit
DMC	Diagnostic monitor configuration
DMSS	Dual-mode subscriber station
GMT	Greenwich mean time
GUI	Graphical user interface
MPU	Modem processing unit
OTA	Over-the-air
QLF	Qualcomm license file
QLMS	Qualcomm license management system
QXDM	Qualcomm extensible diagnostic monitor
UE	User equipment