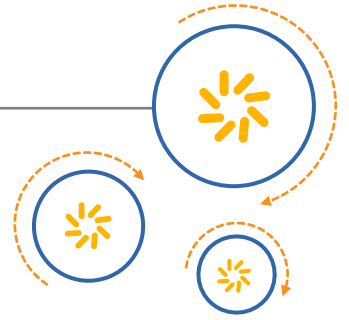




Qualcomm Technologies, Inc.



# QCAT6

## User Guide

80-V1233-6 T

August 24, 2017

**Confidential and Proprietary – Qualcomm Technologies, Inc.**

**NO PUBLIC DISCLOSURE PERMITTED:** Please report postings of this document on public servers or websites to:  
[DocCtrlAgent@qualcomm.com](mailto:DocCtrlAgent@qualcomm.com).

**Restricted Distribution:** Not to be distributed to anyone who is not an employee of either Qualcomm Technologies, Inc. or its affiliated companies without the express approval of Qualcomm Configuration Management.

Not to be used, copied, reproduced, or modified in whole or in part, nor its contents revealed in any manner to others without the express written permission of Qualcomm Technologies, Inc.

QXDM Professional is a product of Qualcomm Technologies, Inc. Other Qualcomm products referenced herein are products of Qualcomm Technologies, Inc. or its subsidiaries.

Qualcomm is a trademark of Qualcomm Incorporated, registered in the United States and other countries. QXDM Professional is a trademark of Qualcomm Incorporated, registered in the United States and other countries. Other product and brand names may be trademarks or registered trademarks of their respective owners.

This technical data may be subject to U.S. and international export, re-export, or transfer ("export") laws. Diversion contrary to U.S. and international law is strictly prohibited.

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

## Revision history

Revision	Date	Description
A	May 2011	Initial release
B	August 2011	Updated Section 3.4.20
C	January 2012	Updated Sections 3.4.8, 3.4.20, and 3.4.25
D	June 2012	Updated Sections 3.4.25.4 and 4.2.1.2
E	August 2012	Updated Sections 3.4.10, 4.2.1.1, 4.2.1.2, 4.2.2.1; Table 4-1; removed nonapplicable information
F	July 2013	Updated Section 3.4.11.1 and Table 4.1
G	August 2013	Updated Table 1-1
H	February 2014	Updated Sections 3.4.22.1, 3.4.22.2, 3.4.22.3, 3.4.22.6, 3.4.22.8, 3.4.23, 3.4.23.1, 4.2.1.1, 4.2.1.2, 4.2.2.1, 4.2.2.2, and Table 4-1
J	April 2014	Updated Sections 1.1, , 3.6.22.1, 3.6.22.2, 3.6.22.6, 3.6.22.8, 3.6.23, 3.6.23.1, 3.6.28.15, 5.2.1.1, 5.2.1.2, 5.2.2.1, 5.2.2.2, and Table 5-1; added Sections 3.3, 3.6.31, 5.2.4 and Chapters 4 and 6 <b>Note:</b> Numerous figures have been updated to reflect changes in the QCAT UI
K	January 2015	Updated Chapter 6 and 7 titles and Sections 3.6.31.1 and 6.24; added Chapter 5 and Section 3.6.31.11.3.
L	March 2015	Updated Sections 3.6.1, 3.6.22.7, 3.6.31.11.3, 6.2.1.1, and Chapter 5
M	April 2015	Updated numerous screenshots and Sections 5.1.1 and 5.2.1; added Section 5.2.1.1
N	June 2015	Updated Sections 5.4 and 5.4.1; added Chapter 7
P	July 2016	Updated Sections 2.1, 3.6.1, 3.6.11.1, 3.6.29.4, 3.6.29.9, 6.2.2.2, and 6.2.4; added Section 3.6.29.14
R	February 2017	Updated Sections 3.6.20.1, 3.6.20.2, 3.6.20.3, 3.6.20.6, 3.6.20.7, 3.6.21, 3.6.25.17, 5.3, 5.3.1, and 7.2.1.1
T	August 2017	Updated Sections 3.6.20.3 and 3.6.20.6

**Note:** There is no Rev. I, O, Q, S, X, or Z per Mil. standards.

# Contents

---

<b>1 Introduction.....</b>	<b>7</b>
1.1 Purpose.....	7
1.2 Conventions .....	7
1.3 Technical assistance.....	7
<b>2 Installing Software.....</b>	<b>8</b>
2.1 QCAT installation.....	8
2.2 License file.....	9
<b>3 QCAT Application .....</b>	<b>10</b>
3.1 Getting started.....	10
3.2 QCAT window views .....	10
3.3 QCAT window views .....	11
3.3.1 Left pane .....	11
3.3.2 Right pane .....	16
3.3.3 Bottom pane.....	18
3.4 QCAT toolbar .....	19
3.5 QCAT status bar .....	20
3.6 Using the QCAT application .....	20
3.6.1 Opening a log file .....	20
3.6.2 Viewing packet contents.....	22
3.6.3 Viewing packet hex dump .....	22
3.6.4 Sorting the display .....	23
3.6.5 Hiding/unhiding log packets.....	23
3.6.6 Additional Info column.....	23
3.6.7 Go to a specific packet.....	26
3.6.8 Setting a time window .....	26
3.6.9 Filtering packet types.....	28
3.6.10 Bookmarks.....	29
3.6.11 Find feature .....	30
3.6.12 Saving parsed text.....	32
3.6.13 Saving in .dlf format .....	33
3.6.14 Saving PPP information in a text file.....	33
3.6.15 Converting a log file to PCAP format .....	34
3.6.16 Converting a PCAP file to .txt format .....	36
3.6.17 Converting a log file to PCAP and .txt format.....	37
3.6.18 Config TShark/Tethereal configuration options .....	38
3.6.19 Merge with Wireshark/Tethereal tool.....	40
3.6.20 Vocoder extraction/playback .....	42
3.6.21 Perl sample scripts .....	49

3.6.22 Call flow analysis support .....	50
3.6.23 Viewing the log summary .....	56
3.6.24 Saving the log summary .....	57
3.6.25 Configuration tab .....	58
3.6.26 Invoking the QCAT user guide.....	64
3.6.27 Viewing the Supported Logs list .....	64
3.6.28 Getting QCAT version information .....	65
3.6.29 Analysis features.....	66
<b>4 Common Analyzers .....</b>	<b>86</b>
4.1 Debug messages vs. time .....	86
4.2 Events and debug messages vs. time .....	87
4.3 Log file information .....	87
4.4 Log mask selection .....	87
4.5 Log packet summary .....	88
4.6 Events vs. time .....	88
<b>5 Command Line Operations .....</b>	<b>89</b>
5.1 Text parsing .....	89
5.1.1 Text parsing options .....	89
5.2 Analysis output .....	92
5.2.1 Analysis output options .....	92
5.3 Vocoder.....	95
5.3.1 Vocoder options.....	95
5.4 ConvertQMDL.....	96
5.4.1 QShrink hash file directory option.....	99
5.5 Convert to .dlf.....	99
5.5.1 .dlf conversion options.....	99
5.6 Convert to .isf .....	100
5.6.1 .isf conversion options .....	100
<b>6 Scripting with QCAT (Windows Only).....</b>	<b>101</b>
6.1 Sample scripts.....	101
6.2 QCAT automation objects .....	102
6.2.1 QCAT6.Application object.....	102
6.2.2 QCAT6.AutoLogPacket .....	113
6.2.3 QCAT6.AutoPacketFilter .....	115
6.2.4 QCAT6.AutoWorkspace .....	116
<b>7 Scripting with QCAT (Mac and Linux Only).....</b>	<b>119</b>
7.1 Sample scripts.....	119
7.2 QCAT automation objects .....	120
7.2.1 QCAT6Application object.....	120
7.2.2 LogPacket .....	132
7.2.3 PacketFilter .....	134
7.2.4 Workspace .....	135
<b>8 QCAT Excel Workbook (Windows Only).....</b>	<b>138</b>

8.1 Viewing the QCAT workbook.....	138
8.1.1 Getting started.....	138
8.1.2 QCAT workbook index page.....	139
8.1.3 QCAT workbook study sheet .....	140
8.1.4 QCAT navigator .....	140
<b>A References.....</b>	<b>143</b>
A.1 Related documents .....	143
A.2 Acronyms and terms .....	143

## Figures

Figure 5-1 Sample path name of an analyzer.....	93
Figure 5-2 Workspace Filter .....	94
Figure 7-1 Example for a path for the SelectOutput method .....	136

## Tables

Table 4-1 Common informational analyzers .....	86
Table 6-1 Listing of sample scripts.....	101
Table 7-1 List of sample scripts.....	119
Table 8-1 QCAT navigator functions .....	140

# 1 Introduction

---

## 1.1 Purpose

QCAT is an integrated software package that allows you to decode the contents of binary log files generated by the Qualcomm Technologies, Inc. (QTI) Mobile Diagnostic Monitor (MDM), CDMA Air Interface Tester (CAIT), and Qualcomm Extensible Diagnostic Monitor Professional™ (QXDM Pro) tools. QCAT is compatible with Microsoft Windows 2000, XP, Vista, and 7 platforms.

QCAT supports performance log packets, mobile events, and logged signaling messages generated by QTI Dual Mode Subscriber Software (DMSS) and Advanced Mode Subscriber Software (AMSS) targets. These include packets related to platforms, such as IS-95, IS-2000 Rel 0, 1xEV-DO, WCDMA, GSM, GPRS, LTE, TD-SCDMA, gpsOne, and Bluetooth®.

For up-to-date information on what packets are supported in specific versions of QCAT, see the Supported Logs listing file. In QCAT, this file can be opened by selecting Help→Supported Logs in the application window.

This document is written for users of QCAT and assumes that the user is familiar with QTI logging concepts, as well as at least one of the Diagnostic Monitor (DM) applications.

## 1.2 Conventions

Function declarations, function names, type declarations, and code samples appear in a different font, e.g., `#include`.

Code variables appear in angle brackets, e.g., `<number>`.

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

Button and key names appear in bold font, e.g., 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/>.

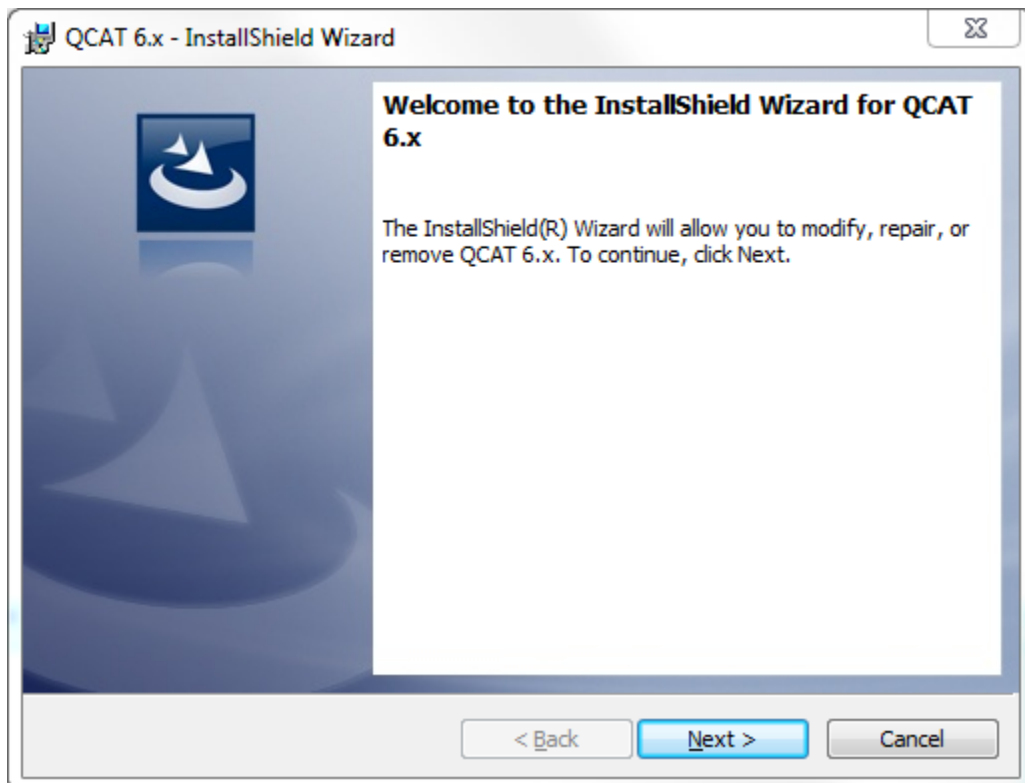
If you do not have access to the CDMATech Support website, register for access or send email to [support.cdmatech@qti.qualcomm.com](mailto:support.cdmatech@qti.qualcomm.com).

## 2 Installing Software

---

### 2.1 QCAT installation

The QCAT installation package has 2 files. One is a watermarked .png file and the other is a Windows MSI installer file with a naming convention of QCAT.xx.xx.xx.msi, where xx.xx.xx represents the major, minor, and build version numbers of QCAT. If the .png file doesn't have the correct user name and product version then QCAT displays a pop-up a error message. Double-click the .msi file to install QCAT. The Windows installer wizard walks you through each step required to install the application. The installer allows you to choose the path to install the application. The QCAT installation setup screen is shown in the following screenshot.



To maintain backward compatibility for automation setups dependent upon tool versions, QCAT 6.x does not require that any previous installation of QCAT 5.x be uninstalled. This is only true for different major versions. Multiple minor revisions of QCAT 6.x will not coexist.



## 2.2 License file

QCAT is distributed with a limited period license. The validity and duration of the license is indicated in the license.txt file that can be found in the same directory as the QCAT executable. The license file can be viewed using a text editor, such as Notepad. It contains a readable Start Date and End Date followed by two or more lines of encrypted data.

To illustrate, the following license file enables QCAT to run from January 10, 2014 to July 10, 2014:

```
File: License.txt
Start Date: 2014 01 10
End Date: 2014 07 10
99 75 55 b9 21 fd 87 81 08 0e 0e db 83 e4 c9 40
d9 02 b7 72 0c fb a7 03 08 dd 0e 9b 83 dc c9 40
```

Making changes to the license file will render it invalid. If you need to renew your license after expiration, contact QTI; see Section [1.3](#).

The QCAT application displays brief license information in its main window on the status bar in the lower right corner. There is usually no need to open the license file directly

# 3 QCAT Application

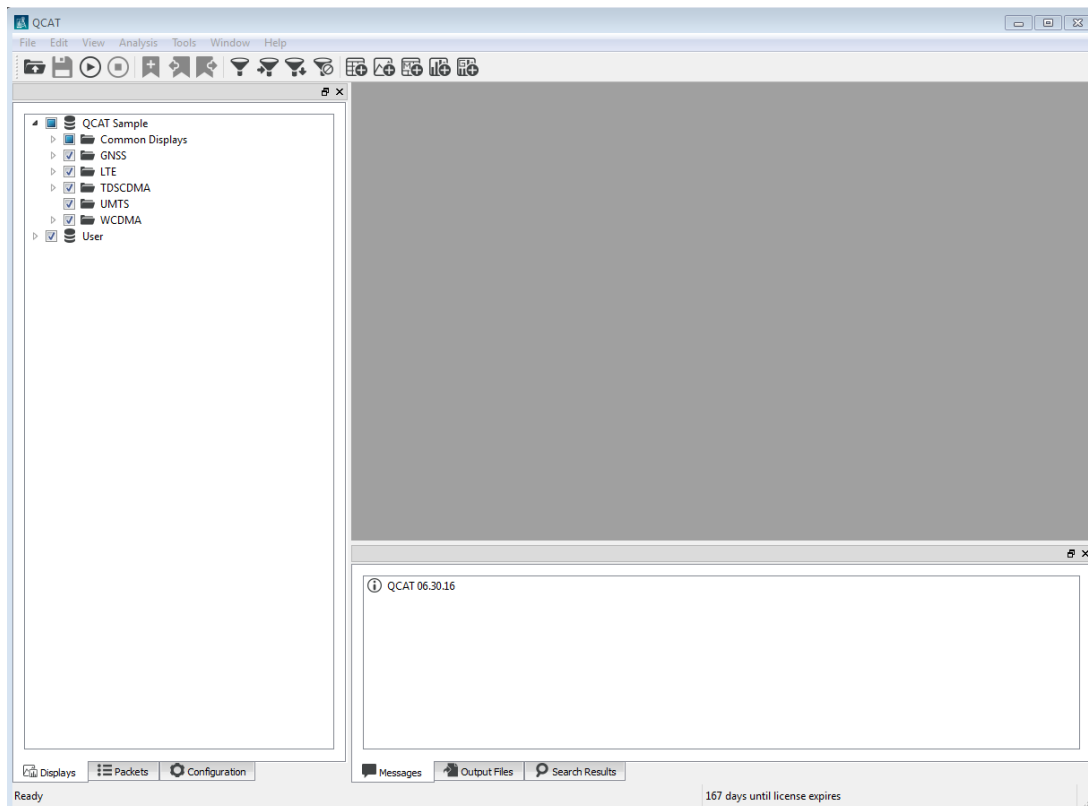
---

## 3.1 Getting started

QCAT can run from the Start→All Programs menu. If you have QXDM installed, you can also launch QCAT from the Tools menu of QXDM. In a typical installation, QCAT can be launched by selecting Start→All Programs→QCAT6→QCAT.

## 3.2 QCAT window views

The main screen of the QCAT application is shown in the following screenshot.



QCAT uses a splitter-view user interface with the following views:

- Workspace/Log packet list/Configuration (left)
- Parsed text/Tree/Analysis view (right)
- Message/Search Result (bottom)

## 3.3 QCAT window views

QCAT uses a three pane UI; the main window views are:

- Left pane – Tabbed view for access to analyzer displays, log packets, and configuration parameters
- Right pane – Displays the selected item from the left view
- Bottom pane – Tabbed view for messages and exported files

### 3.3.1 Left pane

The left pane currently has three active tabs, i.e., Displays, Packets, and Configuration.

#### 3.3.1.1 Log Packets tab (log viewer)

This tab of the left view, called the Log Viewer, displays a list of all log packets in the currently open log file. When a file is open, the bottom right of the status bar displays the number of packets in the file.

#	Time	Type	Description	Subtitle	Direction	Size	Additional Info
0	2015 Jan 21 18:13:16.314	0x1FFD	Diagnostic Version			161	
1	2015 Jan 21 18:13:16.314	0x1FFC	Annotation			57	
2	2015 Jan 21 18:13:16.324	0xB146	LTE LL1 UL AGC Tx Report			204	IQ Gain Backoff = 0.0 dB
3	2015 Jan 21 18:13:16.324	0xB126	LTE LL1 PDSCH Demapper ...			68	
4	2015 Jan 21 18:13:16.324	0xB130	LTE LL1 PDCCH Decoding R...			76	
5	2015 Jan 21 18:13:16.325	0x1FEB	Extended Debug Message			192	
6	2015 Jan 21 18:13:16.325	0xB111	LTE LL1 Rx Agc Log			184	
7	2015 Jan 21 18:13:16.325	0xB146	LTE LL1 UL AGC Tx Report			204	IQ Gain Backoff = 0.0 dB
8	2015 Jan 21 18:13:16.326	0x1FEA	Diagnostic Request	Timestamp Request		13	
9	2015 Jan 21 18:13:16.326	0x1FEB	Extended Debug Message			96	
10	2015 Jan 21 18:13:16.326	0x1FEB	Extended Debug Message			114	
11	2015 Jan 21 18:13:16.326	0x1FEB	Extended Debug Message			179	
12	2015 Jan 21 18:13:16.326	0xB1A7	LTE MLI Scheduler Log			348	
13	2015 Jan 21 18:13:16.326	0xB146	LTE LL1 UL AGC Tx Report			204	IQ Gain Backoff = 0.0 dB
14	2015 Jan 21 18:13:16.326	0xB126	LTE LL1 PDSCH Demapper ...			68	
15	2015 Jan 21 18:13:16.326	0xB130	LTE LL1 PDCCH Decoding R...			48	
16	2015 Jan 21 18:13:16.326	0xB063	LTE MAC DL Transport Block			412	
17	2015 Jan 21 18:13:16.326	0xB146	LTE LL1 UL AGC Tx Report			204	IQ Gain Backoff = 0.0 dB
18	2015 Jan 21 18:13:16.326	0xB126	LTE LL1 PDSCH Demapper ...			68	
19	2015 Jan 21 18:13:16.326	0xB130	LTE LL1 PDCCH Decoding R...			104	
20	2015 Jan 21 18:13:16.326	0xB14D	LTE LL1 PUCCH CSF			24	
21	2015 Jan 21 18:13:16.326	0xB149	LTE LL1 CSF Spectral Efficie...			420	
22	2015 Jan 21 18:13:16.326	0xB14B	LTE LL1 CSF Spectral Efficie...			76	
23	2015 Jan 21 18:13:16.328	0xB066	LTE MAC UL Buffer Status L...			1584	
24	2015 Jan 21 18:13:16.328	0xB126	LTE LL1 PDSCH Demapper ...			68	
25	2015 Jan 21 18:13:16.328	0xB130	LTE LL1 PDCCH Decoding R...			48	
26	2015 Jan 21 18:13:16.328	0x1FEB	Extended Debug Message			106	

Displays
Packets
Configuration

ready
1434403 Packets; 1 Selected

The following information is displayed about each log packet:

- Sequence number
- Time
- Type (log code)
- Description (log title)

- Subtitle (log subdescription)
- Direction (for signaling messages)
- Size (in bytes)
- Additional information (specified data items from the log)

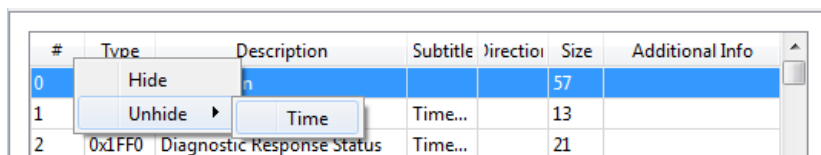
Log packets are initially listed in the same order in which they are read from the log file. Clicking in the column titles, i.e., Type or Size, reorders the packets by the values in those columns.

Clicking a second time in the same column title reverses the order. The number and order of the columns in the display can be customized; for more information, see Section 3.3.1.2. Selecting a log packet entry in this view updates the right view accordingly.

### 3.3.1.2 Column hiding/ordering

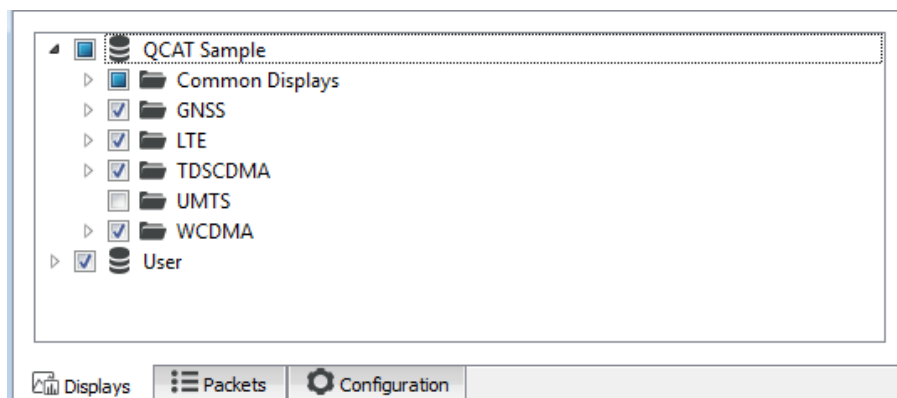
The order and number of columns displayed in the log packets can be modified, as shown below.

- Change column order – Left-click the column header and drag it to the new location.
- Hide a column – Right-click the column header and select Hide.
- Unhide (display) a column – Right-click any column header to get the unhide list; click the column name to bring that column back into the display.

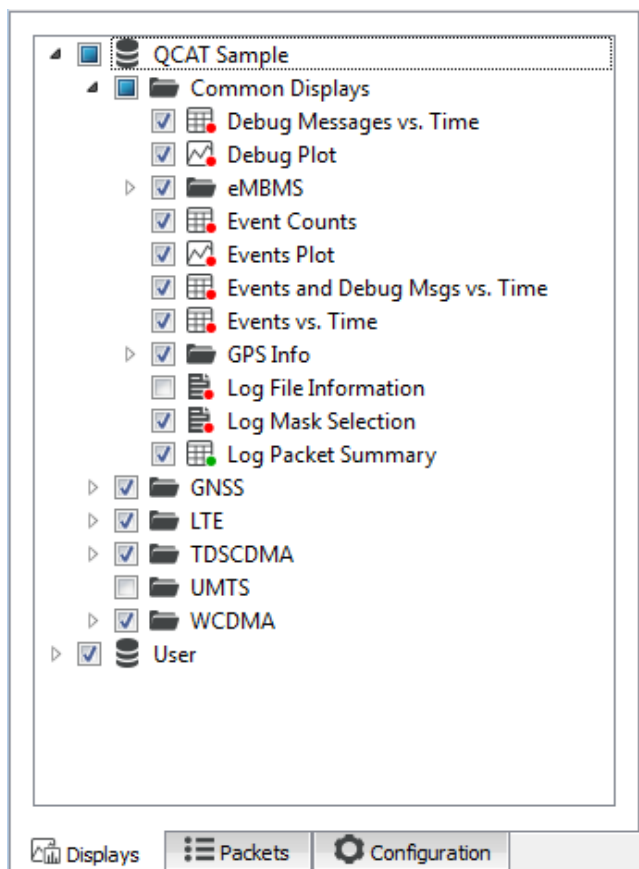


### 3.3.1.3 Analyzer display tab

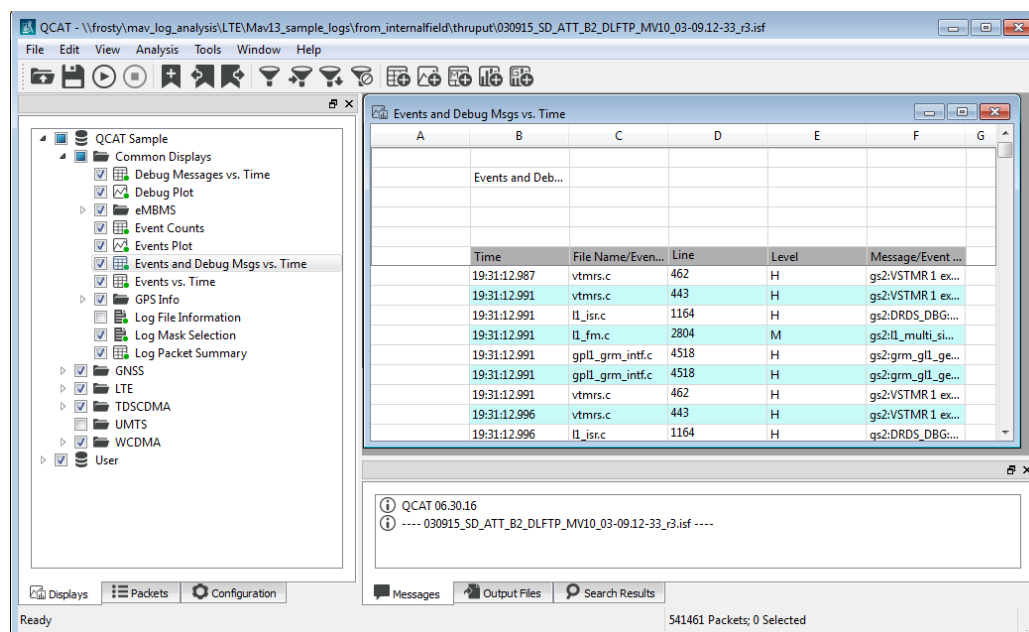
This tab of the left view displays the contents of the current workspace in a hierarchical tree structure as shown.



Clicking a branch of the tree expands it to show the subbranches and individual displays for the summaries, graphs, histograms, plots, and maps that are stored in this workspace.

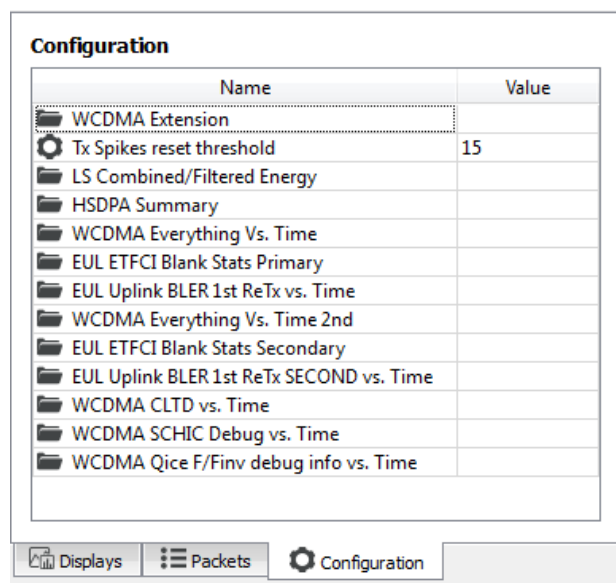


Double-click an entry in the tree to display that analyzer in the right view.



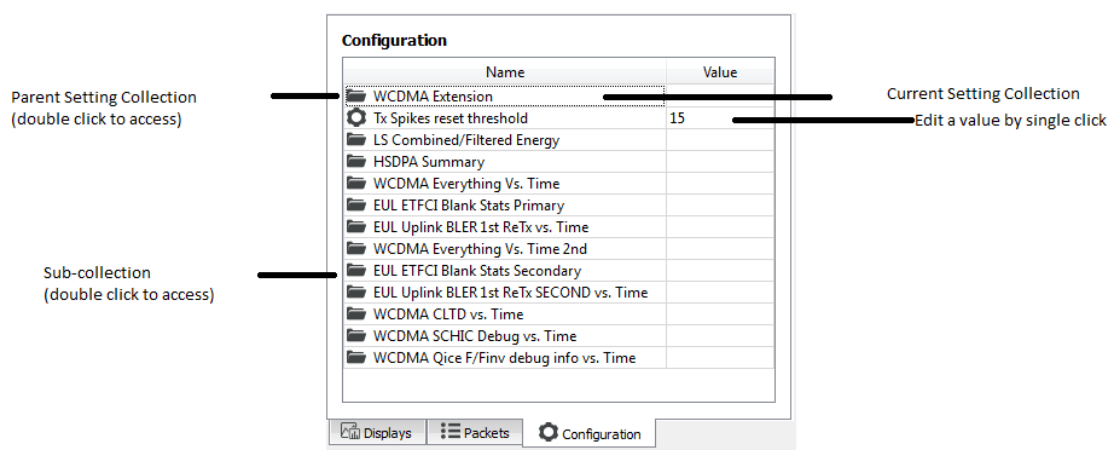
### 3.3.1.4 Configuration tab

Settings for the application and individual analyzers can be accessed through the Configuration tab.



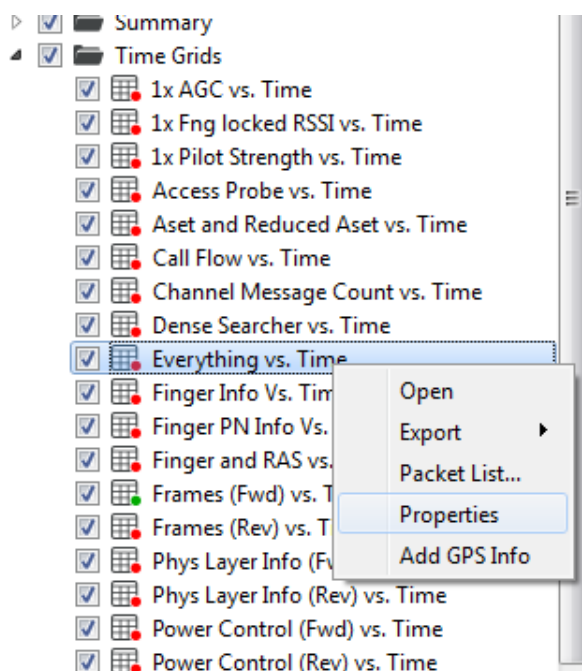
Blue dots mark values that can be edited by single-clicking the line. Folders mark collections of settings that can be accessed by double-clicking.

As shown here, a folder with an arrow pointing up on the first item means that you are looking at a subcollection of settings. The parent scope settings can be accessed by double-clicking this line.



The analyzers, which have configurable properties, can be set from the configuration tab in two ways:

- Navigate through the configuration window to get to the analyzer, thus bringing up the list of properties that can be modified.
- In the analyzer Displays tab, described in Section 3.3.1.3, you can select an analyzer. Right-clicking brings up a contextual menu, where one of the options is Properties. If there are configurable properties for this analyzer, selecting the properties option causes the Configuration tab to display. A property window appears, showing the right-click menu for the Everything vs. Time analyzer, and there is only one configurable property, Resolution.



### Everything vs. Time

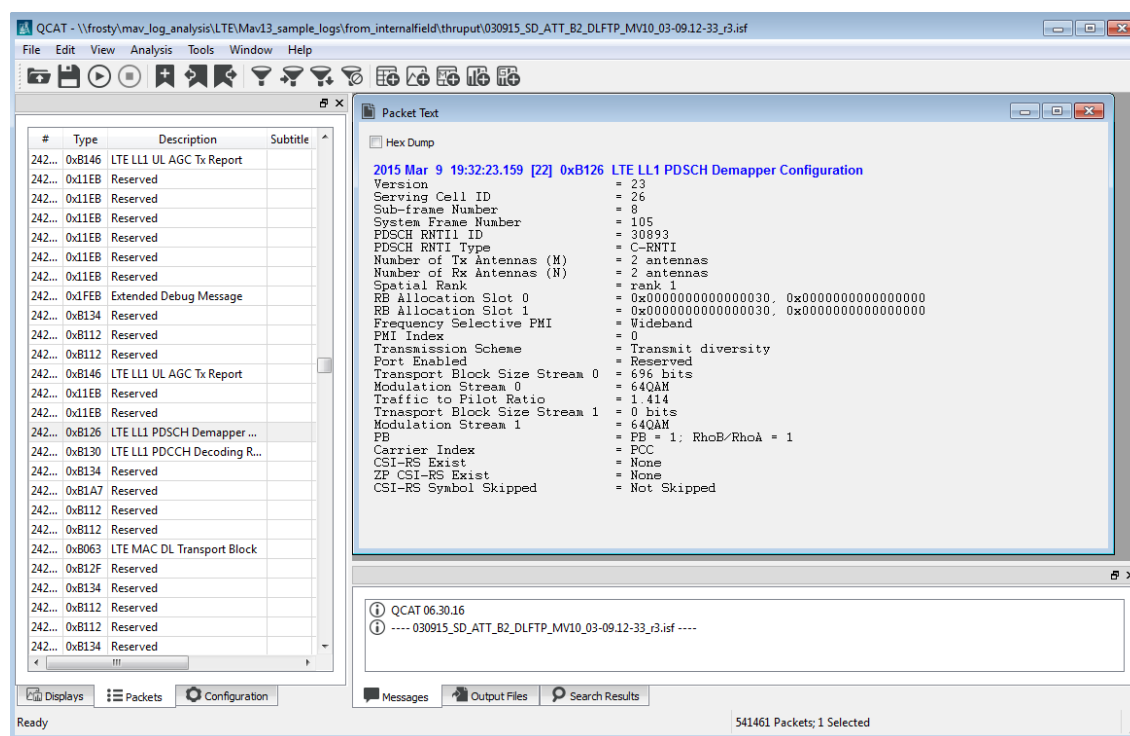
Name	Value
Mobile Everything vs. Time	
Resolution	1

## 3.3.2 Right pane

The right pane holds displays opened from the left pane.

### 3.3.2.1 Log viewer (parsed text)

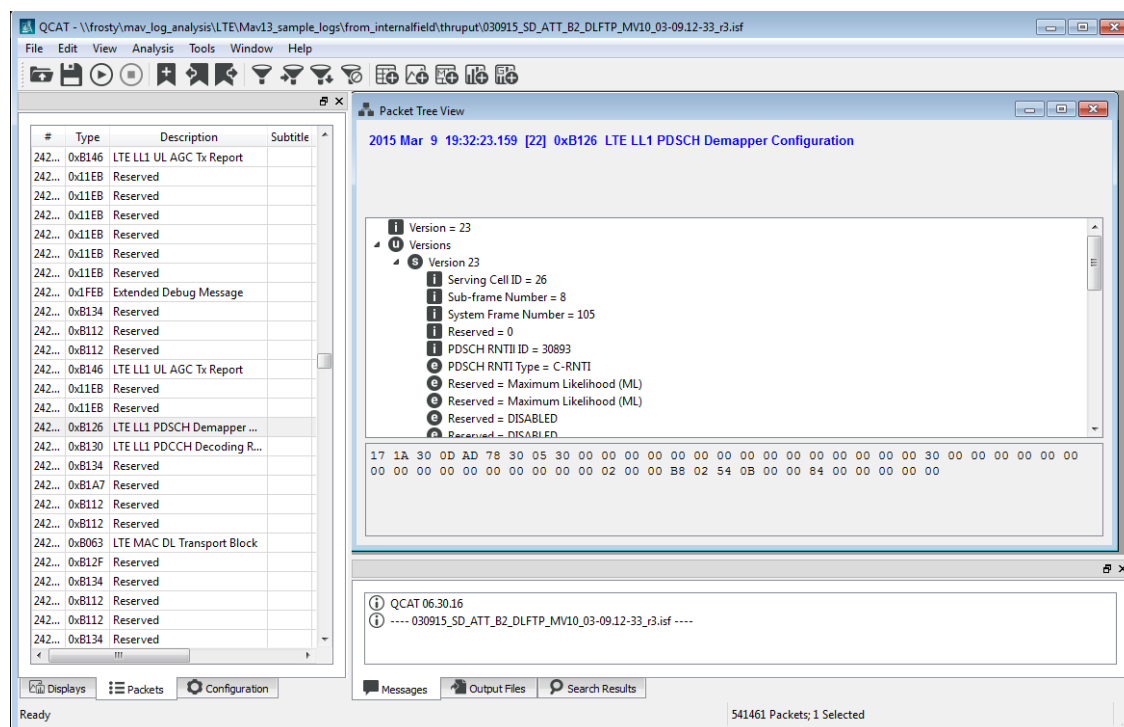
When a log packet is selected from the Log Packets tab in the left view, the packet is parsed and displayed in the right view in the log viewer window, as shown.





### 3.3.2.2 Log viewer (tree view)

Log packets are also parsed according to their structure, which can be seen in the tree view.



The tree view allows the viewing of formulas used for calculated values and bit offsets/lengths of fields. Selecting a field in the tree view results in the bits used to parse it being displayed in red in the hex dump. Also, a description is shown, if available, for each field on the bottom of the window.

The tree view also allows adding fields to the additional information column in the packet list by right-clicking the field and selecting Add to Additional Info column.

### 3.3.2.3 Analyzer

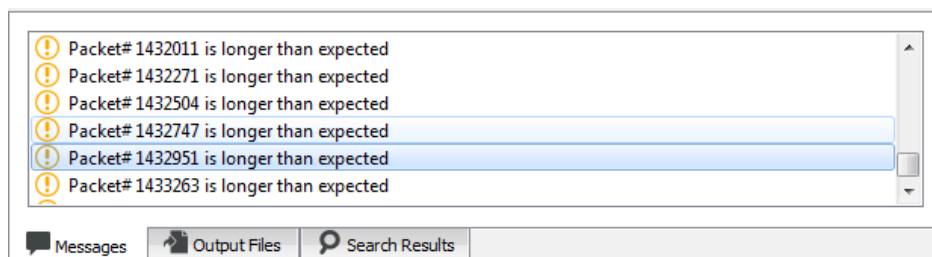
When the displays tab is selected in the left view, the right view displays the selected analyzer.

### 3.3.3 Bottom pane

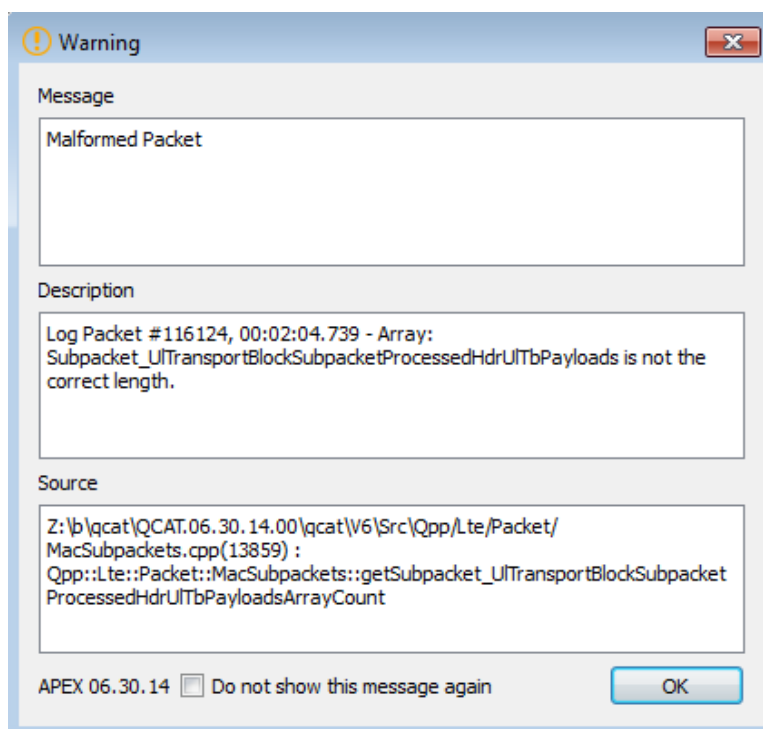
The bottom pane has two tabs, Messages and Exported Files.

#### 3.3.3.1 Messages tab

The Messages tab displays the information, warning, and error messages generated by QCAT. An icon displayed with the message indicates the type of each message.

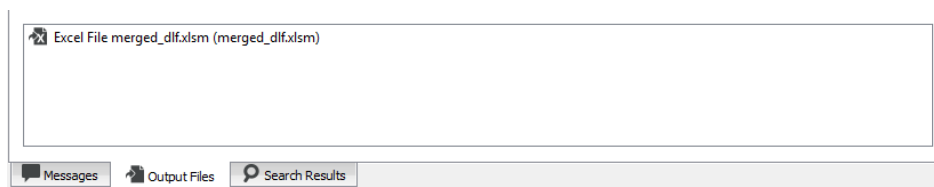


To see a detailed explanation of each message, double-click the message text. This displays a help dialog that explains the context and the source of the message, as shown.

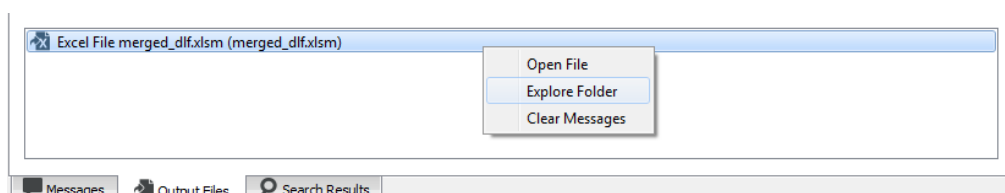


### 3.3.3.2 Exported files tab

The Exported Files tab displays a list of files that QCAT has produced in response to an export or other command. See Section 3.6.29.7 for a description of the export feature.



To view the contents of the text or Excel file, double-click the filename. This opens the file in the associated application. Right-clicking an item displays a contextual menu that displays options to open the file, explore the folder containing the file, or clear the exported files pane.



## 3.4 QCAT toolbar

The QCAT toolbar has a simple set of basic operations.

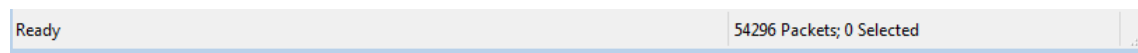


- Open – Starts the open file dialog to open a log file
- Save – Saves the parsed text
- Stop – Stops current parsing or analysis
- Add Bookmark – Toggles whether the selected packets are bookmarked
- Previous Bookmark – Selects the previous bookmarked packet from the list
- Next Bookmark – Selects the next bookmarked packet from the list
- Filter – Opens the filter packet dialog
- Open Filter – Opens the file open dialog for opening a filter file
- Save Filter – Opens the file save dialog to save the current filter
- Clear Filter – Clears current filter entities and resets to original settings
- Add Grid – Opens the dialogs for a new packet time grid
- Add Plot – Opens the dialogs for a new plot based on existing analyzers
- Add Merged Grid – Opens the dialogs for a new merged grid based on existing analyzers
- Add Histogram – Opens the dialogs for creating a new histogram based on a log packet field
- Add Grid Histogram – Opens the dialogs for creating a histogram based on the column of an existing analyzer

## 3.5 QCAT status bar

The QCAT Status bar displays menu status messages, the count of log packets in the currently opened log file, and the license indicator. The license indicator displays the number of remaining days of license validity.

The Status bar also displays the progress bar and progress information during the File→Open and File→Save as text operations.

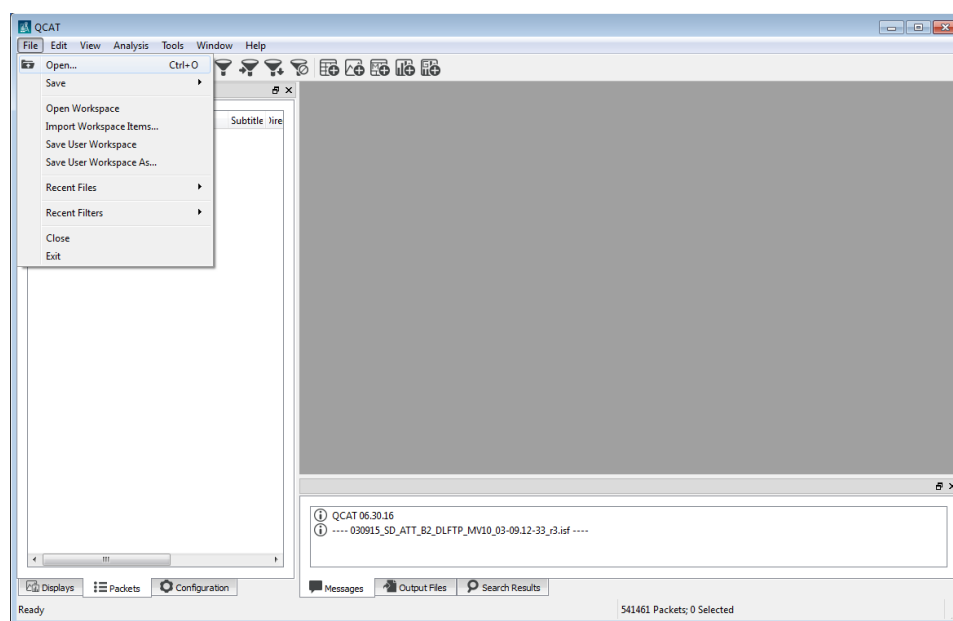


## 3.6 Using the QCAT application

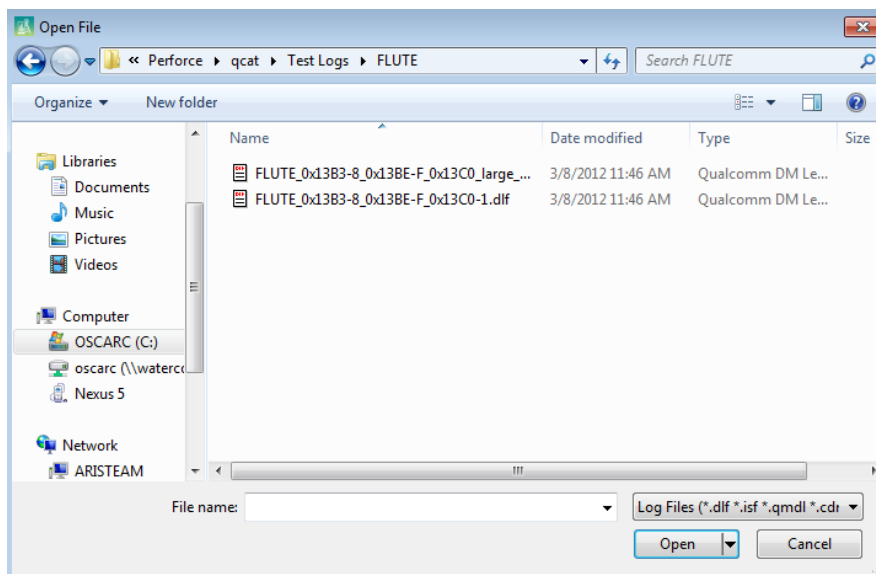
The following sections explain the various features provided by the QCAT application and menus, shortcut keys, and buttons used to invoke them.

### 3.6.1 Opening a log file

To open a log file, select File→Open. The shortcut for this function is Ctrl+O.



The Windows File→Open dialog box used to browse directories for files with extension \*.dlf or .isf appears. To open log files with other extensions, select All Files (\*.\*) in the Files of type drop-down list. Files must be in binary format to be readable by QCAT.



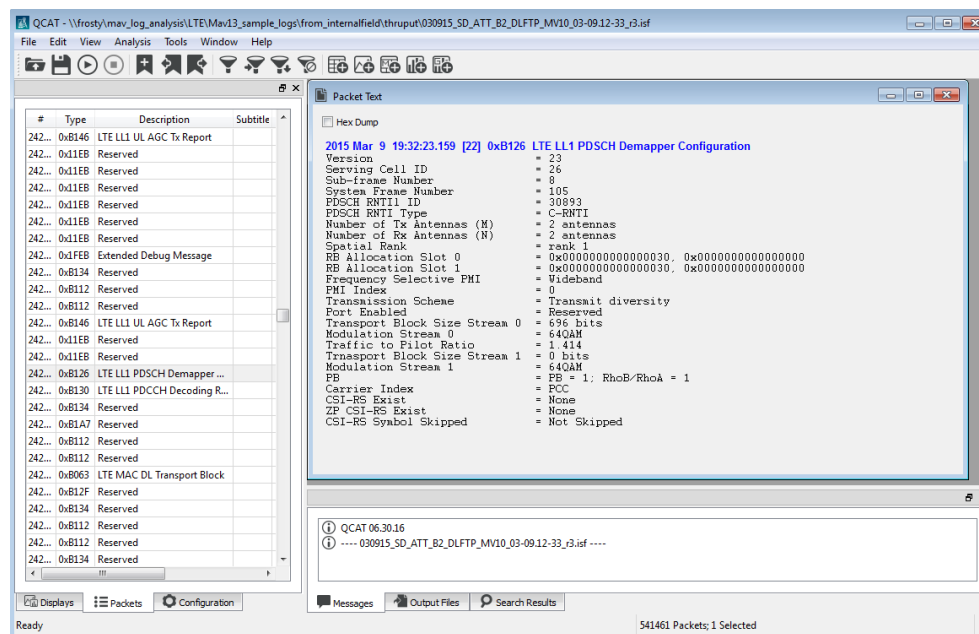
QCAT also provides a Most Recently Used (MRU) file list in the File menu. This allows you to select the last four log files from the File menu and open them directly.

On most Windows systems, double-clicking a .dlf file automatically launches QCAT and opens the file.

**NOTE:** If opening a .qmdl or .qmdl2 file QCAT automatically searches the directory and any subdirecotries for QShrink4 hash files (\*.qdb or \*.qsr4) to use when decoding QShrink messages.

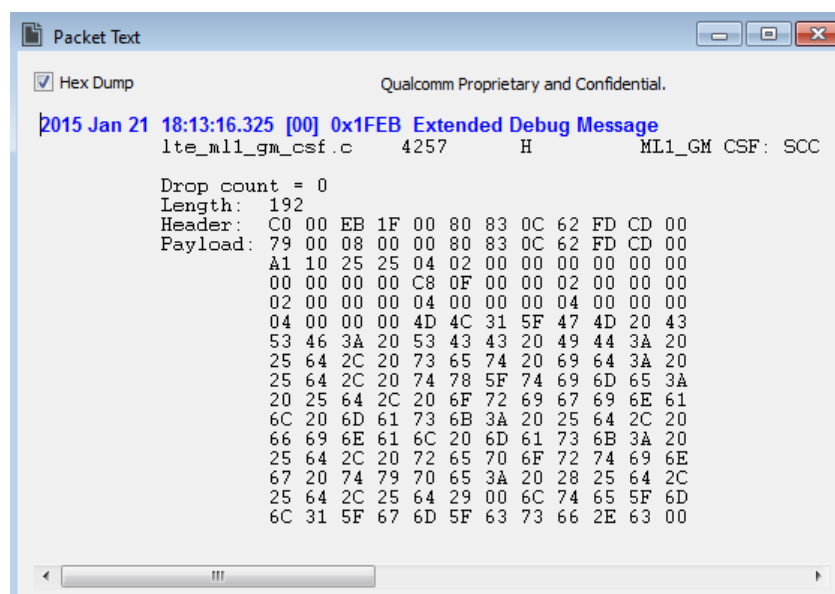
### 3.6.2 Viewing packet contents

To parse and view the contents of any log packet listed in the left view, click that entry. QCAT parses the selected log packet and displays the text and or tree view corresponding to the header, as well as payload of the packet, in the right view.



### 3.6.3 Viewing packet hex dump

The right view contains the hex dump of the header and contents for the packet selected in the left view. An example of a hex dump display is shown here.



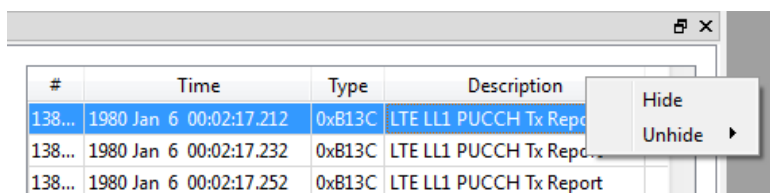
### 3.6.4 Sorting the display

Click any of the column headers in the left view to sort the list by column. Clicking that column header a second time reverses the sorted order.

### 3.6.5 Hiding/unhiding log packets

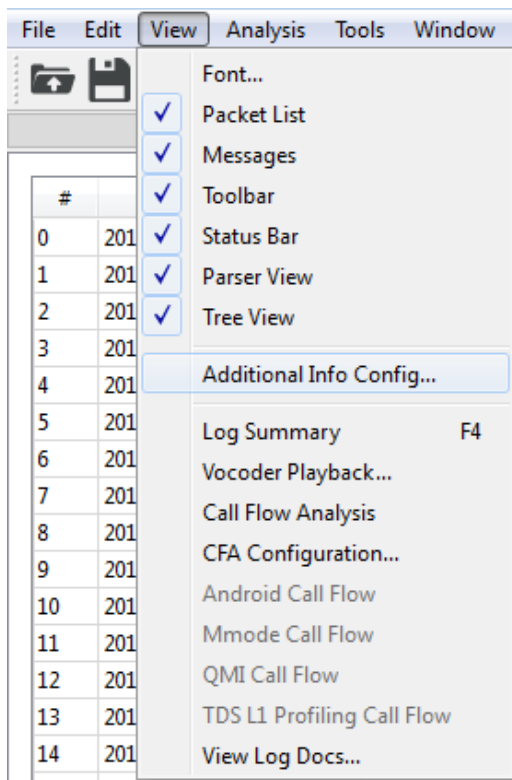
The Hide/Unhide feature can be used to limit the display to certain log packets. To remove log packets from view, select the packets in the left view, then select Edit→Hide Packets from the menu. The shortcut for this function is Ctrl+H.

To make hidden packets visible again, select Edit→Unhide Packets. Select the packets (or click **Select All**), and then click **Unhide**.



### 3.6.6 Additional Info column

The Additional Info column can be customized to show data from specific fields in different log packets. This can be achieved by selecting the fields from the tree view and adding them to the column, or by selecting View→Additional Info Config.



If fields selected for the Additional Info column are not shown, they do not exist in that instance of the log packet. This could be due to several factors, which include, but are not limited to:

- The packet is malformed.
- The field is in an array and the index is larger than the maximum size of the instance.
- The field belongs to a different version of the packet.
- The field is optional and does not exist in the specific instance.

An example of the Additional Info column is shown here.

Description	ibit	Direction	Size	Additional Info
Diagnostic Version			161	
Annotation			57	
LTE LL1 UL AGC Tx Report			204	IQ Gain Backoff = 0 dB
LTE LL1 PDSCH Demapper ...			68	
LTE LL1 PDCCH Decoding R...			76	
Extended Debug Message			192	
LTE LL1 Rx Agc Log			184	
LTE LL1 UL AGC Tx Report			204	IQ Gain Backoff = 0 dB
Diagnostic Request	Ti...		13	
Extended Debug Message			96	
Extended Debug Message			114	
Extended Debug Message			179	
LTE ML1 Scheduler Log			348	
LTE LL1 UL AGC Tx Report			204	IQ Gain Backoff = 0 dB
LTE LL1 PDSCH Demapper ...			68	
LTE LL1 PDCCH Decoding R...			48	

### 3.6.6.1 Additional Info configuration dialog

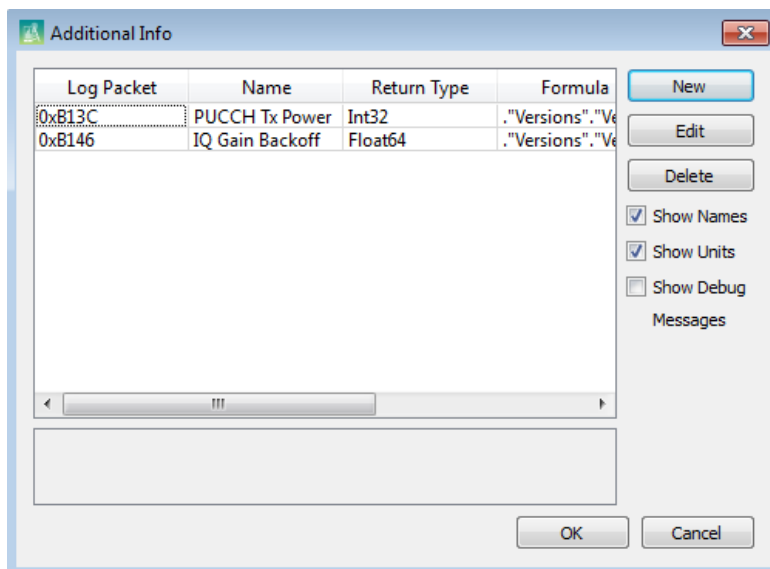
The Additional Info dialog displays all items that are currently populated in the Additional Info column. The dialog gives a quick summary of the:

- Log Packet – Log for which the item applies
- Name – Name of the field to be displayed
- Return Type – Type to be displayed in the column
- Formula – Formula used internally to retrieve the value; this follows the log structure from the base to the field
- Unit – Unit of the data, e.g., dB, sec, etc.

From the dialog, new items can be added and existing items can be edited or deleted. Selecting New or Edit invokes the Additional Info Item dialog, as described in Section 3.6.6.2.

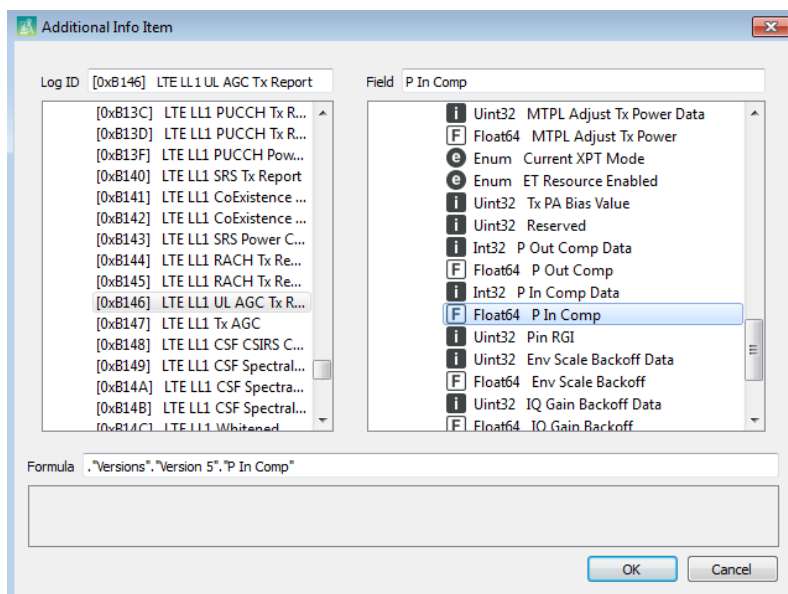


To save space, the Show Name and Show Unit options disable printing of the name or unit in the Additional Info Column.



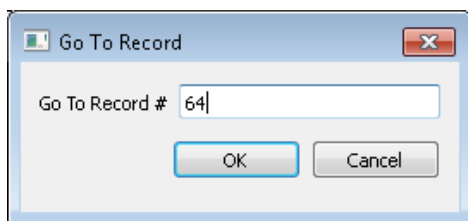
### 3.6.6.2 Additional Info Item dialog

The Additional Info Item dialog allows the selection of fields to show in the Additional Info column. The dialog includes a list of packets for which fields can be chosen. Selecting a packet causes that packet's structure to be displayed in the Log Structure view. Selecting a field automatically populates the Formula Edit box and the description. The formula can then be edited to set appropriate array indices.



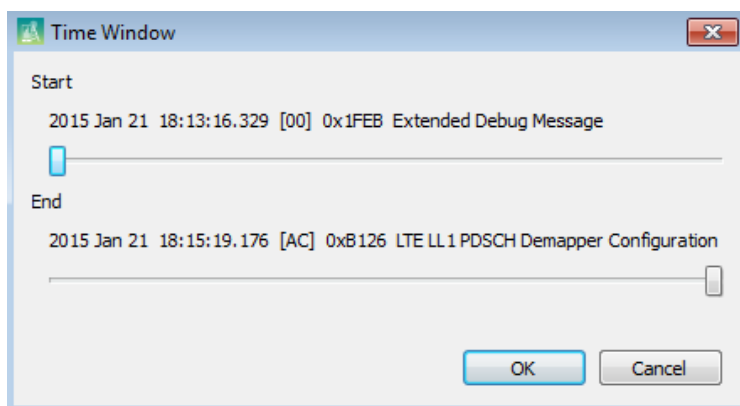
### 3.6.7 Go to a specific packet

This feature allows jumping to a packet for a given record number. Select Edit→Go To Packet to bring up the dialog box.



### 3.6.8 Setting a time window

This feature allows hiding/unhiding of packets based on a window defined by a start and end time. Select Edit→Time Window, then set the start and end times for the window by moving the slider controls.

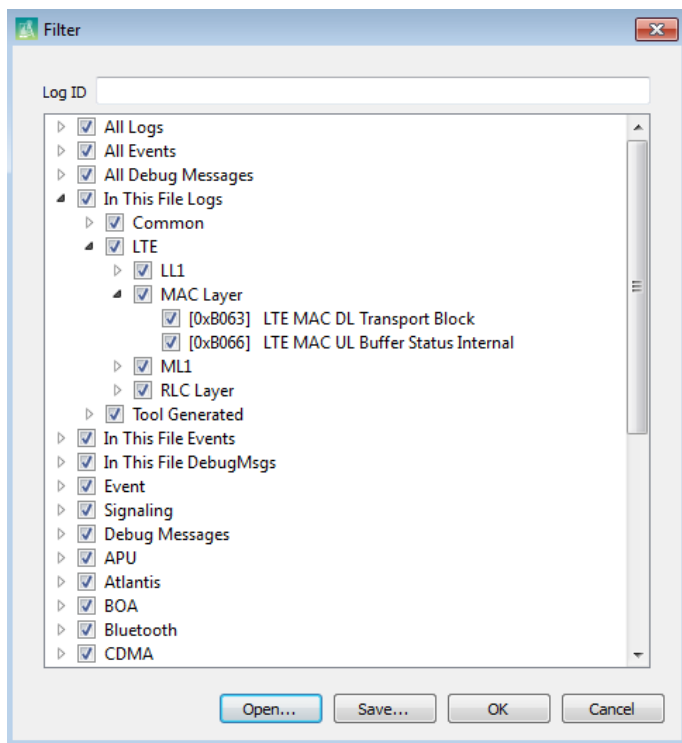


The start and end times can also be set from the packet list, by right-clicking a packet and selecting “Begin Time Window Here” or “End Time Window Here.” The options are shown in the following screenshot.

#	Time	Type	Description	Subtitle
28	2015 Jan 21 18:13:16.329	0x1FEB	Extended Debug Message	
29	2015 Jan 21 18:13:16.329	0xB146	LTE LL1 UL AGC Tx Report	
30	2015 Jan 21 18:13:16.329	0xB126	LTE LL1 PDSCH Demapper ...	
31	2015 Jan 21 18:13:16.329	0xB130	LTE LL1 PDCCH Decoding R...	
32	2015 Jan 21 18:13:16.330	0xB146	LTE LL1 UL AGC Tx Report	
33	2015 Jan 21 18:13:16.330	0xB126	LTE LL1 PDSCH Demapper ...	
34	2015 Jan 21 18:13:16.370	0x1FF0	Diagnostic Response Status	Timestamp Response
35	2015 Jan 21 18:13:16		Hide	
36	2015 Jan 21 18:13:16		Unhide	ling R...
37	2015 Jan 21 18:13:16		Hide Similar Packets	age
38	2015 Jan 21 18:13:16		Hide Similar Events	age
39	2015 Jan 21 18:13:16		Show Only Similar Packets	Version Information ...
40	2015 Jan 21 18:13:16		Show Only Similar Events	age
41	2015 Jan 21 18:13:16		Begin Time Window Here	age
42	2015 Jan 21 18:13:16		End Time Window Here	age
43	2015 Jan 21 18:13:16		Time Window...	port
44	2015 Jan 21 18:13:16		Bookmark	age
45	2015 Jan 21 18:13:16		Add Note...	oper ...
46	2015 Jan 21 18:13:16		Remove Bookmark	ling R...
47	2015 Jan 21 18:13:16		Show Only Bookmarked	age
48	2015 Jan 21 18:13:16			age
49	2015 Jan 21 18:13:16.331	0x1FEB	Extended Debug Message	

### 3.6.9 Filtering packet types

This feature allows hiding/unhiding of packets based on type code. Access this feature by selecting Edit→Filter.



The filter is applied to the file currently open and to any other files that are opened during the same QCAT session. If the filter is set before opening a log file, filtered packets cannot be unhidden and the file must be reopened with a different filter to get the missing packets. If no log file is open, the filter shows all supported packet types. Clicking Select All checks (keeps) all listed packet types; clicking Unselect All unchecks (discards) all listed packet types.

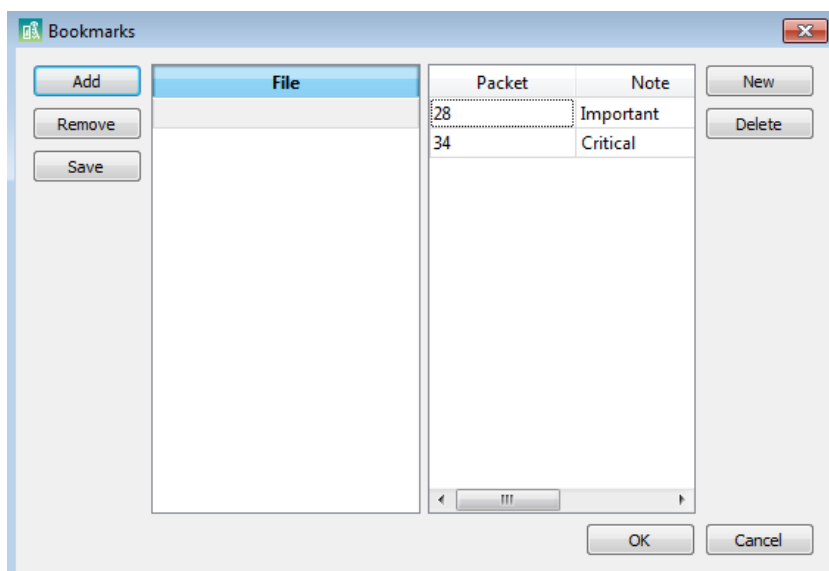
### 3.6.10 Bookmarks

QCAT allows bookmarking packets for easy navigation between significant packets. When a packet is bookmarked, the index is shown in a light color.

#	Time	Type	Description	Subtitle
28	2015 Jan 21 18:13:16.329	0x1FEB	Extended Debug Message	
29	2015 Jan 21 18:13:16.329	0xB146	LTE L1 UL	
30	2015 Jan 21 18:13:16.329	0xB126	LTE L1 PD	
31	2015 Jan 21 18:13:16.329	0xB130	LTE L1 PD	
32	2015 Jan 21 18:13:16.330	0xB146	LTE L1 UL	
33	2015 Jan 21 18:13:16.330	0xB126	LTE L1 PD	
34	2015 Jan 21 18:13:16.370	0x1FF0	Diagnostic	
35	2015 Jan 21 18:13:16.330	0xB113	LTE L1 PSS	
36	2015 Jan 21 18:13:16.330	0xB130	LTE L1 PD	
37	2015 Jan 21 18:13:16.331	0x1FEB	Extended D	
38	2015 Jan 21 18:13:16.331	0x1FEB	Extended D	
39	2015 Jan 21 18:13:16.331	0x1FEA	Diagnostic	
40	2015 Jan 21 18:13:16.331	0x1FEB	Extended D	
41	2015 Jan 21 18:13:16.331	0x1FEB	Extended D	
42	2015 Jan 21 18:13:16.331	0x1FEB	Extended D	
43	2015 Jan 21 18:13:16.331	0xB146	LTE L1 UL	
44	2015 Jan 21 18:13:16.331	0x1FEB	Extended Debug Message	

Hide  
Unhide  
Hide Similar Packets  
Hide Similar Events  
Show Only Similar Packets  
Show Only Similar Events  
Begin Time Window Here  
End Time Window Here  
Time Window...  
Bookmark  
Add Note...  
Remove Bookmark  
Show Only Bookmarked

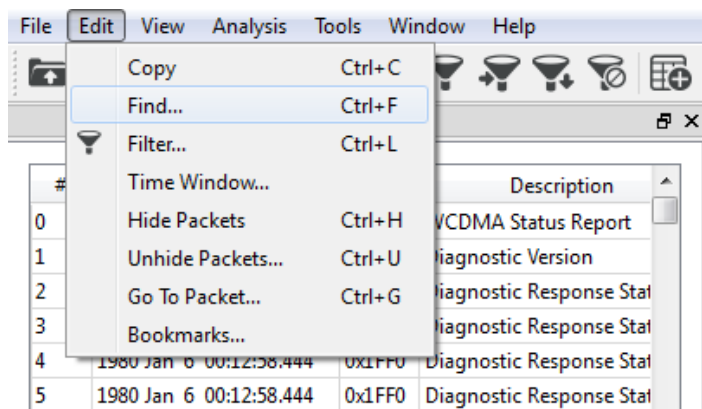
To bookmark a packet, either right-click the packet and select Bookmark, or add bookmarks from the Bookmark dialog available from Edit→Bookmarks.



The bookmark dialog allows opening and saving files containing bookmark information so that bookmark files can be passed along with log files for quick analysis. Multiple files can be opened simultaneously to show comments from multiple users. Bookmarks from each file are displayed as a different color. It also provides a way to add notes to bookmarks to explain why a packet is significant. This note displays when the cursor hovers over the index in the packet list and is printed in the header of the packet in the parsed view.

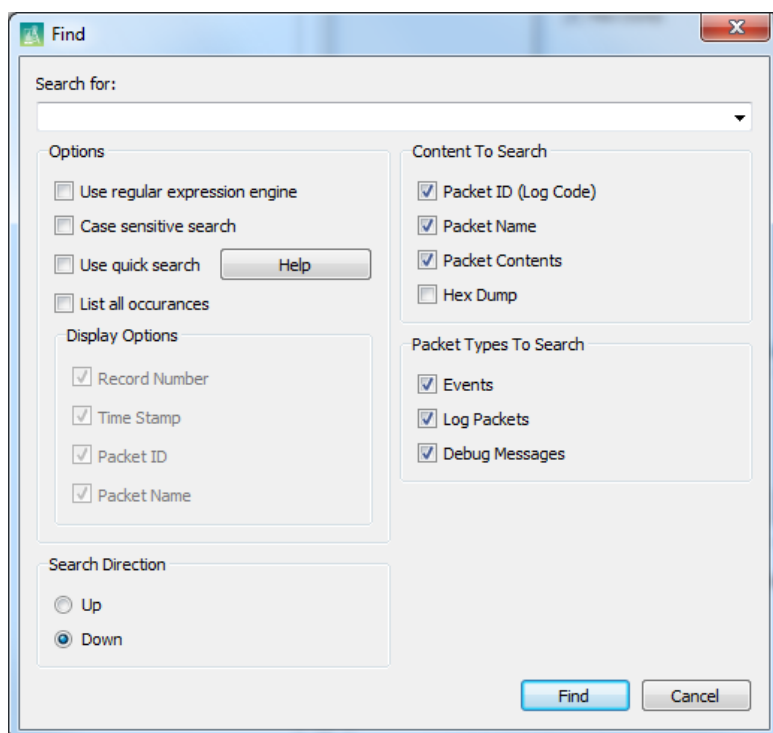
### 3.6.11 Find feature

This feature can be used to search for text or a regular expression within the parsing output. Select Edit→Find from the menu. The shortcut for this function is Ctrl+F.



#### 3.6.11.1 Find options

The Find operation can be fine-tuned, as shown here. By default, the search is case-insensitive, the search direction is down, and all three packet types (events, regular log packets, and debug message packets) are searched. Also by default all three types of packet content (ID, name, and contents) are searched. As an option, the hex dump can also be searched.

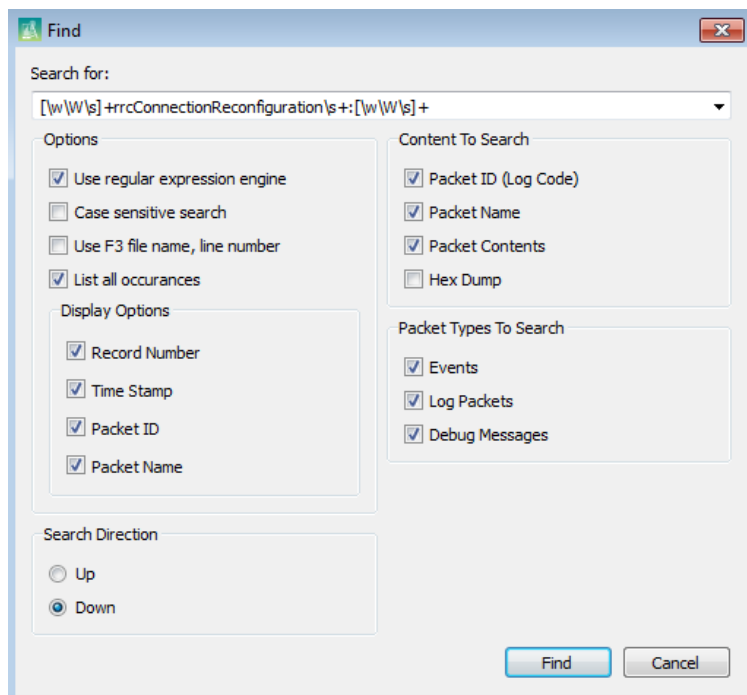


When the text has been found, the search can be repeated by pressing F3 to find the next occurrence. Pressing Shift+F3 repeats the search but for the previous occurrence.

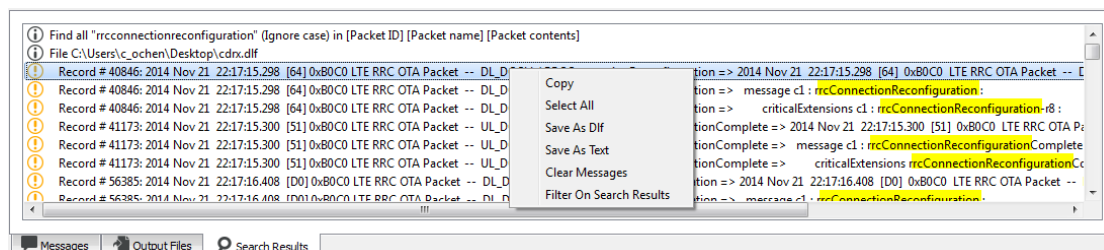
The parser accepts regular expressions when the **Use Regular Expression Engine** option is checked. Documentation on regular expression syntax is beyond the scope of this document.

The Quick Search option allows fields or debug messages to be quickly searched based on names, but it cannot match actual data. This option allows searching quickly between different instances of text.

Global search can be enabled by checking the **List all occurrences** option. Its Search Display Options will be also enabled, as shown here. The **Regular Expression Engine** option is supported with the global search.

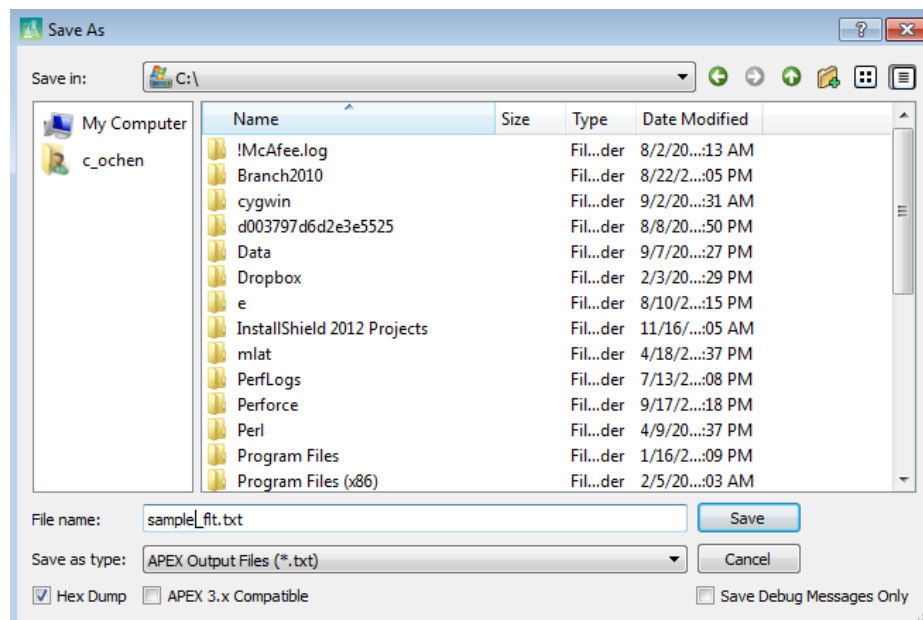


Once the search is done, all matched results will be displayed in the Search Results tab. To jump to the desired packet for details, just double-click the corresponding result record. If further action is required to process search results, right-click the Search Results pane or the desired result record to clear, copy, or save results to a .txt/.dlf file. As result records can be saved to a .dlf file, the global search can be used as a filter based on packet contents.



### 3.6.12 Saving parsed text

To save parsed information for the entire log file to a text file, select File→Save→Text, or press Ctrl+S to bring up the Save As dialog box.



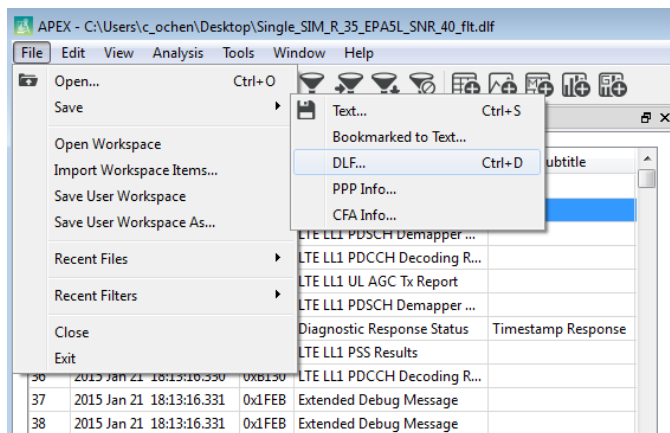
The Save As dialog allows you to save the parsed text to a directory and filename of your choice. The dialog also displays some options that allow you to customize the format of the saved text file:

- **Hex Dump** – In addition to the packet contents, QCAT also saves the hex dump for each packet.
- **QCAT 3.x Compatible** – The text file is saved in the format used by QCAT Ver 3.x and earlier. This option is usually used when you have automation scripts that depend on the QCAT 3.x text output file format.
- **Save Debug Messages Only** – This filters out and saves only the packets that contain debug messages.



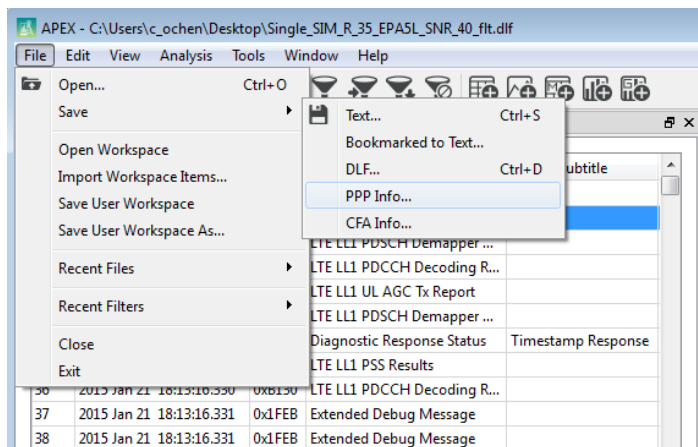
### 3.6.13 Saving in .dlf format

To save the current log file as a .dlf file to a directory and filename of your choice, select File→Save→DLF or press **Ctrl+D**. This feature enables you to save any editing changes you have made to the viewing format of the file, such as sorting and hiding/unhiding logs.

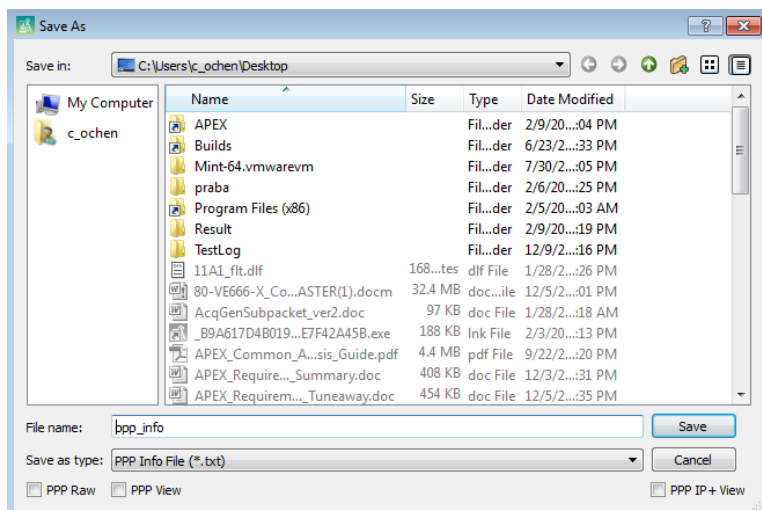


### 3.6.14 Saving PPP information in a text file

To save the PPP information for Tx/Rx frames in the current log file as a text file to a directory and filename of your choice, select File→Save→PPP Info.

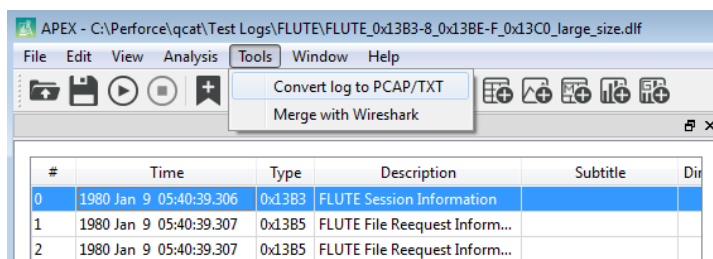


A Save As dialog box appears. At the bottom of the box, select one or more of the four different views of the PPP data that should be saved in the text file.



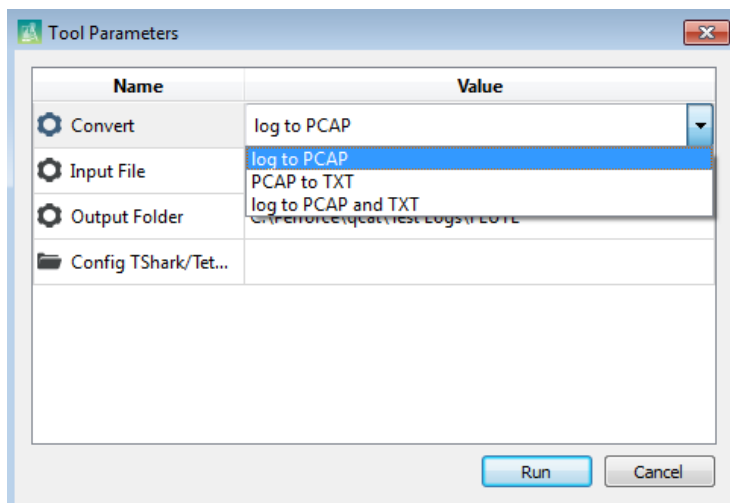
### 3.6.15 Converting a log file to PCAP format

To convert a log file (.isf/.dlf) to a PCAP format file, select Tools→Convert log file to PCAP/TXT.



Select log to PCAP in Convert. Select the log file to convert by selecting the Input File option and clicking **Browse** to locate the file.

Select the output folder by selecting the Output Folder option and clicking **Browse** to locate the folder. Click **Run**. The log file is converted to PCAP files, which are written to the output folder and listed in the Exported Files pane.



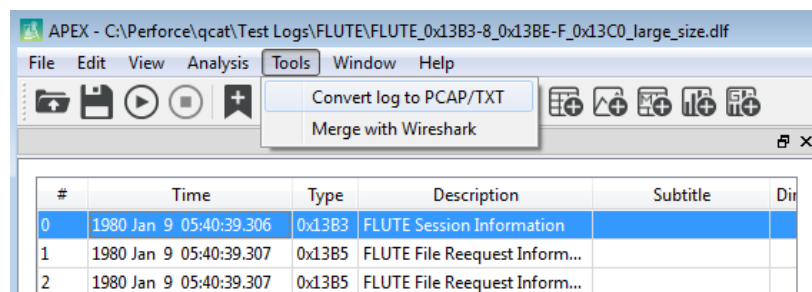
### 3.6.15.1 Support for Wireshark and Ethereal

Wireshark and Ethereal are both supported. Wireshark takes priority over Ethereal. TShark and Tethereal are text versions of Wireshark and Ethereal respectively. There is no option to select Ethereal if both Wireshark and Ethereal are installed on the system. Since Wireshark is the most current, it is recommended. A migration message displays if only Ethereal is found on the system. Download and install Wireshark from <http://www.wireshark.org/download.html>.

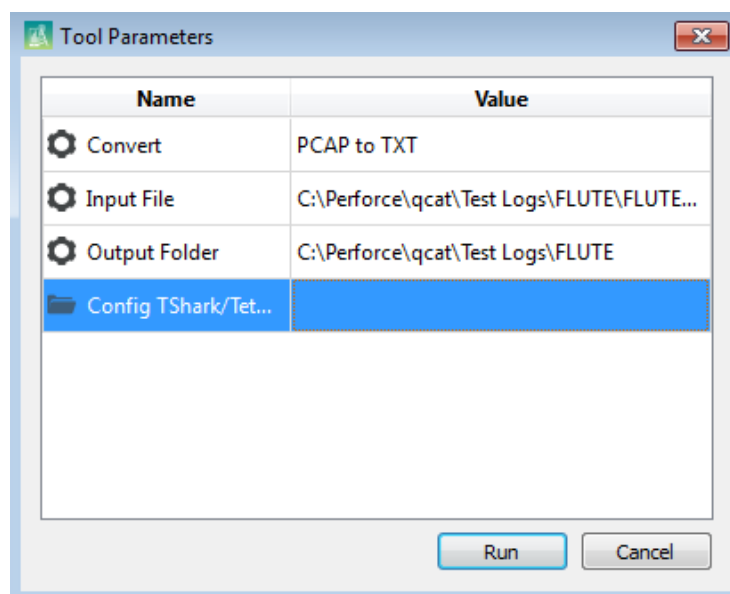
### 3.6.16 Converting a PCAP file to .txt format

To convert a PCAP file to a .txt format file, select Tools→Convert log file to PCAP/TXT. Note that either Wireshark or Ethereal must be installed prior to executing this feature.

See Section 3.6.15.1 for information on Wireshark and Ethereal.



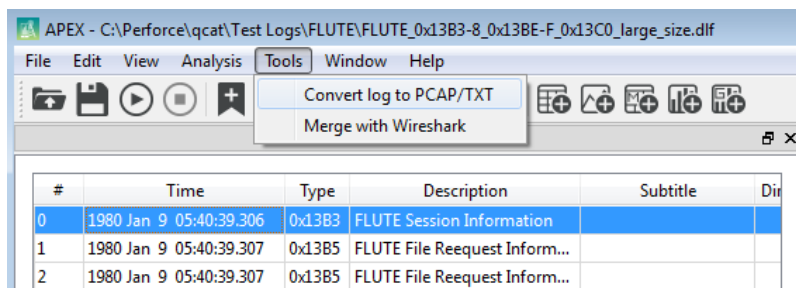
Select PCAP to TXT in Convert, then select the PCAP file to convert by selecting the Input File option and clicking **Browse**. By default, it is set to the recently opened file in QCAT. Select the output folder by selecting the Output Folder option and clicking **Browse** to locate the folder. For optional configurations that TShark/Tethereal support, double-click the Config TShark/Tethereal folder.



For optional configurations for .txt file conversion, double-click the Config TShark/Tethereal folder. See Section 3.6.18 for more information.

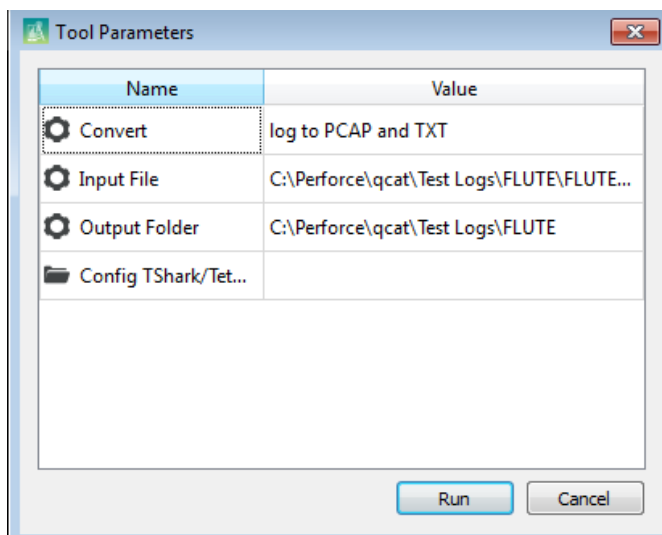
### 3.6.17 Converting a log file to PCAP and .txt format

To convert a log file to PCAP and .txt format files, select Tools→Convert log file to PCAP/TXT. Note that either Wireshark or Ethereal must be installed prior to execution of this feature.



See Section 3.6.19.1 for information on Wireshark and Ethereal.

Select Convert→log to PCAP and TXT, then select the log file to convert by selecting the Input File option and clicking **Browse** to locate the file. Select the output folder by selecting the Output Folder option and clicking **Browse** to locate the folder.



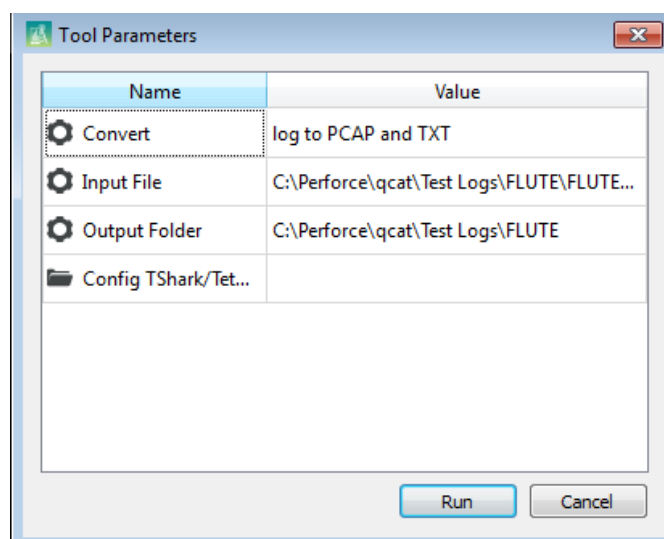
For optional configurations for .txt file conversion, double-click the Config TShark/Tethereal folder. See Section 3.6.19 for more information.

### 3.6.18 Config TShark/Tethereal configuration options

There are three optional configurations for Config TShark/Tethereal described in the following sections.

#### Wireshark/Ethereal filtering

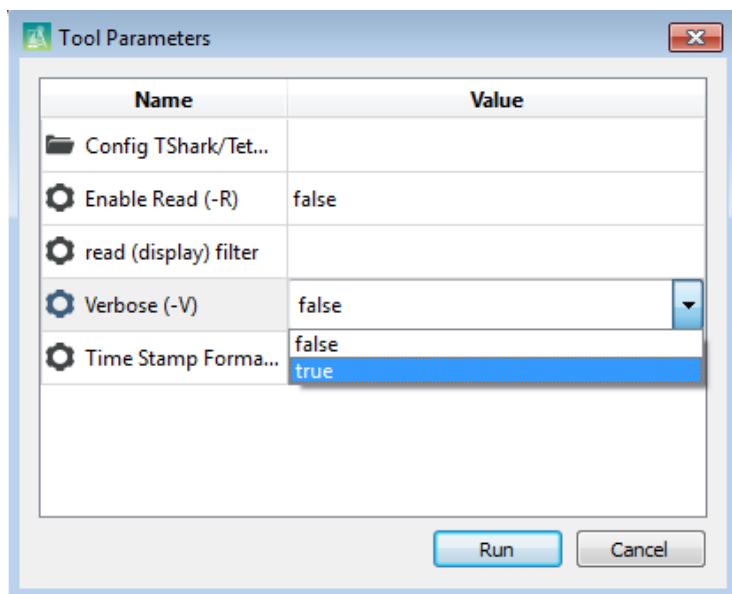
To use a Wireshark/Ethereal filter, set Enable Read (-R) to true and enter the filter in the text field beside the read (display) filter. The filter has exactly the same functionality as in Wireshark or Ethereal. As in Wireshark/Ethereal, double quotes are not needed around the filter value. If a filter is not needed, disable the option by setting Enable Read (-R) to False. If Enable Read (-R) is set to False, even if there are valid filter values in the line below, the result is not filtered. To learn how to use the filter, look for Wireshark or Ethereal filter syntax and reference on the website.



## Verbose option

To use this option, set Verbose (-V) to True. Assuming TShark/Tethereal runs correctly, the PCAP file will be converted to a .txt file, which is written to the output folder and listed in the Exported Files pane.

**NOTE:** Setting the Verbose option to True may dramatically increase the conversion time.



## Timestamp format option

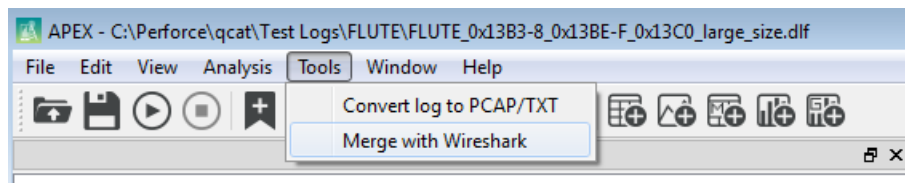
Select the desired timestamp format option from the drop-down list.

- Select *a* for absolute format.
- Select *ad* for absolute with date.
- Select *d* for delta.
- Select *r* for relative; this is the default option.

### 3.6.19 Merge with Wireshark/Tethereal tool

See Section 3.6.19.1 for information on Wireshark and Ethereal.

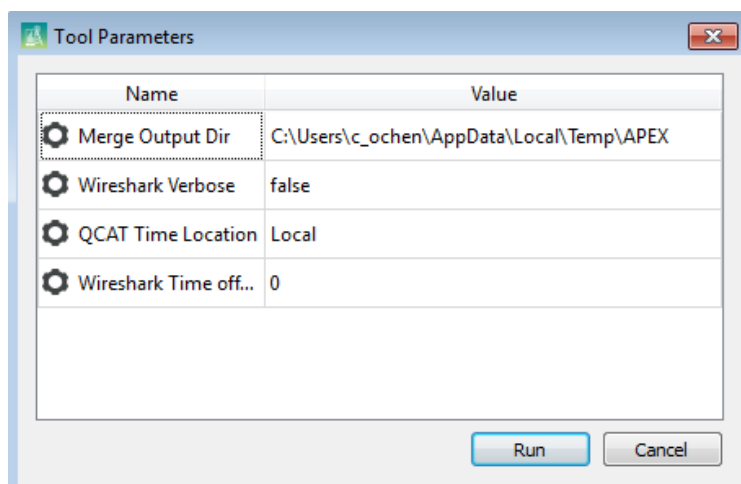
The Merge with Wireshark or Merge with Ethereal functionality is available under the Tools menu in QCAT.



Either function requires an open log file to run:

- If only Wireshark is installed, Merge with Wireshark is available.
- If only Ethereal is installed, Merge with Ethereal is available.
- If both are installed, Wireshark takes priority.
- If none is found, this option shows error text.

The following is the dialog screen for setting merging options.

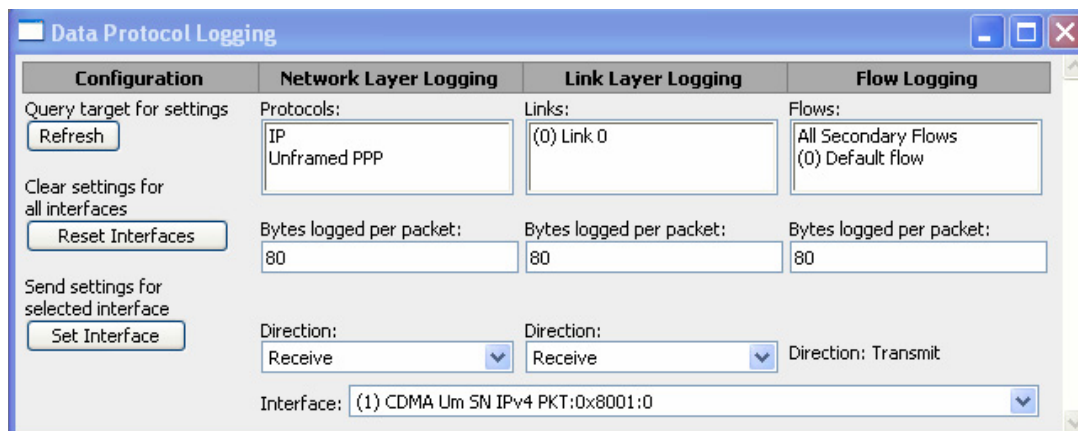


The options are:

- Merge Output Dir – This is an output directory for the merged file. By default, it is set to the QCAT output directory that is available under Configuration.
- Wireshark Verbose – This is an option for TShark/Tethereal Verbose mode to be used. By default, it is set to False.
- QCAT Time Location – This is an option for the TShark/Tethereal .txt files timestamp location. It can be set to Local or UTC.
- Wireshark Time Offset – This is the TShark/Tethereal PPP packets time offset. By default, it is set to 0.



**NOTE:** The Wireshark/Tethereal tool expects PPP packets with at least 3-digit timestamp resolution for a good placement of Tethereal .txt packets in the QCAT-parsed output. This can be done by logging PPP packets at the IP level, instead of at the PPP level. The Data Protocol Logging view in QXDM provides an interface for configuring these logging settings.



### 3.6.19.1 Merge with Wireshark/Tethereal tool from command prompt

The Merge tool is available from the command prompt through the Perl script MergeWithWireshark.pl.

```
perl MergeWithWireshark.pl [--h] [--V] [--time <ARG>] [--offset <ARG>] [--outputDir <ARG>] [--logFile <ARG>] [--pcapFiles <ARG>] [--pgm <ARG>]
```

Options:

--h : Display help

--V : Output TShark/Tethereal txt in Verbose mode, if not present default = summary mode.

--pgm <QCAT> an application to run to generate parsed output( if not present, default = QCAT).

--time <Time location Local/UTC>, if not present default = Local.

--offset <Time offset in ms>, if not present default = 0.

--outputDir <Merged file output dir>, if not present default = c:\Tmp.

--logFile <Full path of dlf/isf/txt files > Where dlf/isf input file from which the parsed output and ppp/pcap binary files will be generated for merging. If the input file is a parsed txt file then --pcapFiles option is must.

--pcapFiles <comma separated list of PCAPGenerated files to merge>

## Examples

- perl merge.pl—h
- perl merge.pl—logFile c:\log.dlf—pgm QCAT—outputDir 'c:\Merged Files'
- perl merge.pl—logFile c:\parsed.txt—pcapFiles 'c:\file1.pcap,c:\file2.pcap'

### 3.6.20 Vocoder extraction/playback

QCAT can extract vocoder frames and PCM samples from log packets and optionally save them as files or play them back.

#### 3.6.20.1 PCM extraction

PCM packets contain the original PCM samples sent to the vocoder (Tx) or the PCM samples coming out of the vocoder (Rx). Based on raw data format being logged with one or more of the following log items, PCM extraction generates .wav (PCM data), .sbc (mSBC data), .cvsd (CVSD data), and/or .bin (bitstream data) files, but the playback function works only on .wav and .raw files. Other file formats can be played via third-party tools, e.g., Audacity application, etc.

- LOG\_TX\_PCM\_C (0x13B0 Ver0) – Rx PCM samples
- LOG\_RX\_PCM\_C (0x13B1 Ver0) – Tx PCM samples
- Wireless Connectivity Audio Data (0x1558) – Rx/Tx PCM, mSBC, and CVSD data
- Audio Vocoder Data Paths (0x13B0 ver1) – Multiple Rx/Tx PCM samples per SessionId and ChannelInterleave
- Audio Vocoder Rx PCM (0x13B1 ver1) – Multiple Rx PCM samples per SessionId and ChannelInterleave
- ADSP Audio Decoder Input Log (0x152E) – Rx PCM stream and raw bitstream
- ADSP Audio Per Object Postprocessing Input Log (0x152F) – Rx PCM stream
- ADSP Audio Rx Matrix Mixer Input Log (0x1530) – Rx PCM stream
- ADSP Audio Common Object Postprocessing Input Log (0x1531) – Rx PCM stream
- ADSP Audio Common Object Preprocessing Input Log (0x1532) – Tx PCM stream
- ADSP Audio Tx Matrix Mixer Input Log (0x1533) – Tx PCM stream
- ADSP Audio Per Object Preprocessing Input Log (0x1534) – Tx PCM stream
- ADSP Audio Encoder Input Log (0x1535) – Tx PCM stream
- ADSP Audio Encoder Output Log (0x1536) – Tx PCM stream and raw bitstream
- ADSP AFE Rx/Tx PCM Log (0x1586) – Rx/Tx PCM stream
- Voice FW Processing Rx/Tx Log (0x158A) – Rx/Tx PCM stream
- Voice FW Stream Rx/Tx Log (0x158B) – Rx/Tx PCM stream

### 3.6.20.2 Vocoder extraction

Vocoder packets contain the actual encoded frames after encoding (Tx) or before decoding (Rx). QCAT can extract these frames and convert them to PCM using one of the C-SIM converters distributed with the application. Vocoder extraction/conversion/playback works only if one or more of the following packets are logged:

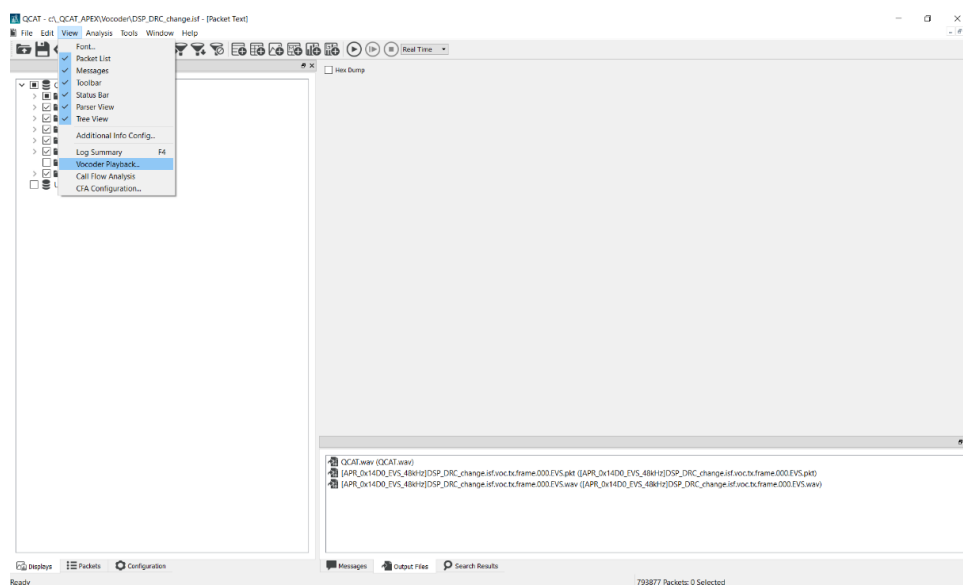
- LOG\_WCDMA\_UL\_TX\_VOCODER\_INFO (0x713F)
- LOG\_WCDMA\_DL\_RX\_VOCODER\_INFO (0x7140)
- LOG\_UMTS\_UL\_TX\_VOCODER\_INFO\_V2 (0x7143)
- LOG\_UMTS\_DL\_RX\_VOCODER\_INFO\_V2 (0x7144)
- LOG\_VOC\_FOR\_TYPE (0x1009)
- LOG\_VOC\_REV\_TYPE (0x100A)
- LOG\_APR\_MODEM (0x14D0)
- LOG\_APR\_ADSP (0x14D2)
- LOG\_ADSP\_CVD\_TX (0x1804)
- LOG\_ADSP\_CVD\_RX (0x1805)
- LOG\_MODEM\_VOICE\_SERVICE\_UL (0x1914)
- LOG\_MODEM\_VOICE\_SERVICE\_DL (0x1915)

The following vocoders are currently supported:

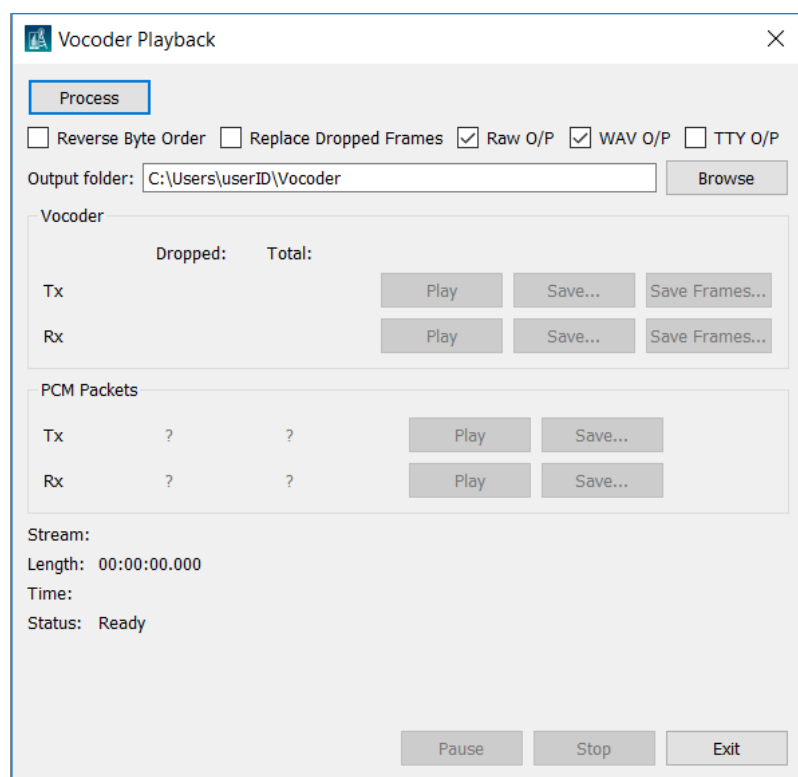
- 13K (Qualcomm Code Excited Linear Prediction 13 kbps)
- EVRC (Enhanced Variable Rate CODEC)
- EVRC-B (Enhanced Variable Rate CODEC B)
- EVRC-WB (Enhanced Variable Rate CODEC Wideband)
- EVRC-NW (Enhanced Variable Rate CODEC, Service Option 73)
- EVRC-NW2K (Enhanced Variable Rate CODEC, Service Option 77)
- HR (GSM Half-Rate)
- FR (GSM Full-Rate)
- EFR (GSM Half-Rate)
- AMR[-NB] (Adaptive Multirate [Narrowband])
- AMR-WB (Adaptive Multirate Wideband)
- EAMR (Enhanced Adaptive Multirate)
- EVS (Enhanced Voice Services)
- G711

### 3.6.20.3 Vocoder control

The vocoder playback feature is available after a log file has been opened. To access the vocoder playback feature, select **View > Vocoder Playback**.



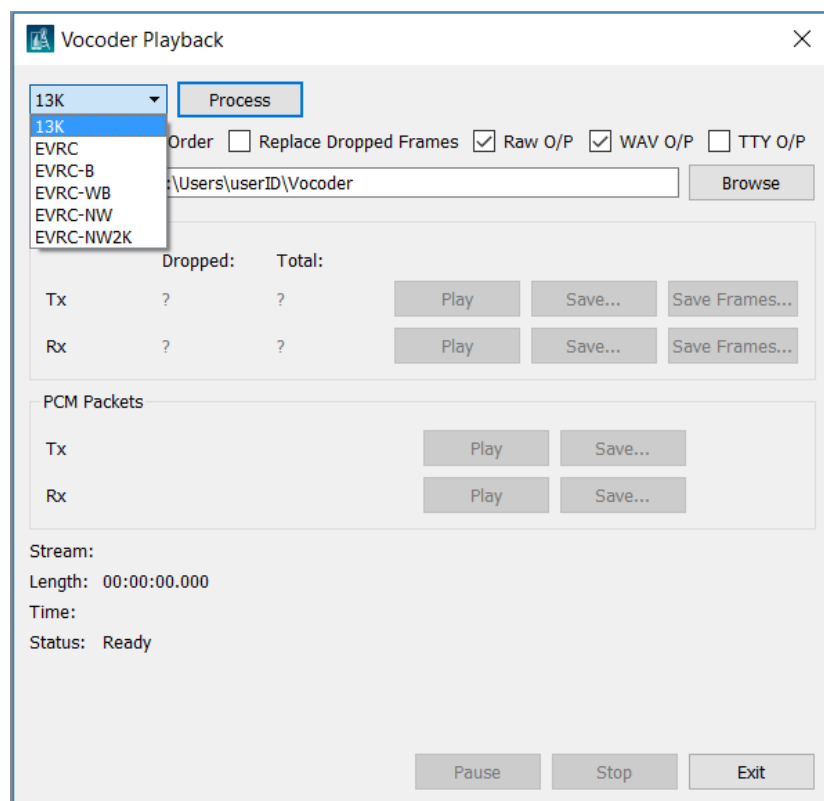
NOTE: The following graphics have been updated.



- The vocoder type can be selected from the drop-down list. This option is not necessary if the log file contains any of the following packets:
  - LOG\_UMTS\_UL\_TX\_VOCODER\_INFO\_V2 (0x7143)
  - LOG\_UMTS\_DL\_RX\_VOCODER\_INFO\_V2 (0x7144)
  - LOG\_APR\_MODEM (0x14D0)
  - LOG\_APR\_ADSP (0x14D2)
  - LOG\_ADSP\_CVD\_TX (0x1804)
  - LOG\_ADSP\_CVD\_RX (0x1805)
  - LOG\_MODEM\_VOICE\_SERVICE\_UL (0x1914)
  - LOG\_MODEM\_VOICE\_SERVICE\_DL (0x1915)
- The Reverse Byte Order option (disabled by default) may be needed for some targets. If audio is unintelligible, turn this option on and reprocess the file.
- The Replace Dropped Frames option (disabled by default) configures the processor to fill any time gap with blank frames.
- The TTY O/P option (disabled by default) works mainly for the log points containing TTY signals (Windows only).

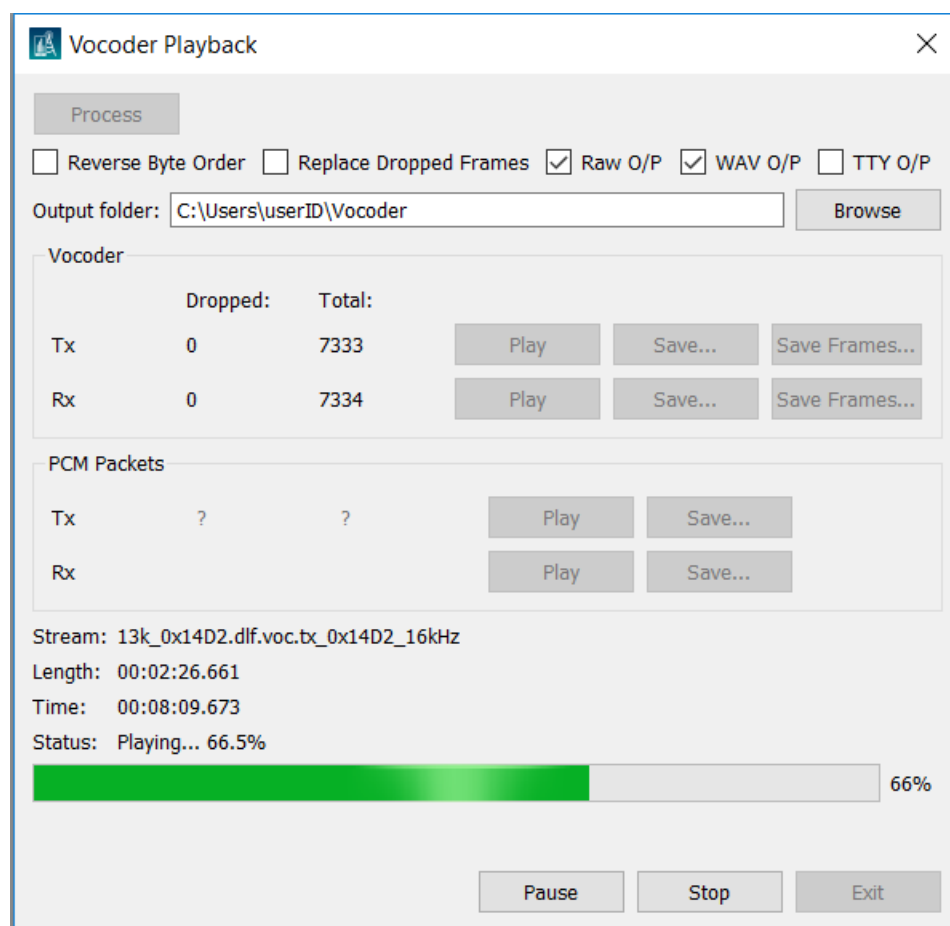
Click **Process** to extract PCM samples and vocoder frames from the log file.

NOTE: The following graphic has been updated.



After reading the log file, the **Play** and **Save** buttons are activated for each stream that has valid data. The “Vocoder” box is for vocoder packet-extracted files, i.e., files in xxx.voc.tx[rx].wav format extracted from vocoder packet log codes, such as 0x7143, etc., and the “PCM packets” box is for PCM packet-extracted files, i.e., files in xxx.pcm.tx[rx].wav format extracted from PCM packet log codes, such as 0x13B0, etc. To play a stream, click **Play** for that channel. The “Stream” field reports to which stream you are listening. The “Length” field reports the length of time for the chosen stream, and the “Time” field reports the approximate log file timestamp that corresponds to the current playback position. The stream can be paused or stopped, and other UI features are available while the stream is playing. To listen to the other stream, click **Play** for the desired stream. If no frames for that stream are available, the status field reads “Play Failed.”

NOTE: The following graphic has been updated.



### 3.6.20.4 Saving generated PCM files from vocoder packets

The **Save** button from the Vocoder box for a stream opens a dialog to save the audio stream as a raw PCM file.

### 3.6.20.5 Saving generated PCM files from PCM packets

The **Save** button in the PCM packets box for a stream opens a Browse for Folder dialog to save multiple audio streams as raw PCM files.

### 3.6.20.6 Filename formats and extensions

- Generated files from PCM packets logged:
  - 0x1558 – Wireless connectivity audio data; multiple Tx/Rx PCM/CVSD/mSBC files  
 \*.pcm.<AudioSource>.<dir>.wav (.cvsd, .sbc)  
 <AudioSource> mapping (default is NA):
    - 4096 (0x1000) – BTSpkr
    - 4097 (0x1001) – BTMic
    - 4098 (0x1002) – BTA2DP
    - 4099 and 4100 (0x1003 and 0x1004) – FM
  - 0x13B0 Ver1 – Audio vocoder data paths; multiple Tx/Rx PCM files  
 \*.pcm.<sessionId>.<ChannelInterleave>.<dir>.wav  
 <ChannelInterleave> mapping:
    - 0x0100 and 0x0101 – INTERNAL\_AFE
    - 0x0102 and 0x0103 – AUXPCM\_AFE
    - 0x0104 and 0x0105 – I2S\_AFE
    - 0x0106 and 0x0107 – MI2S\_AFE
    - 0x0108 – HDMI\_AFE
    - 0x010B – DIGITAL\_MIC
    - 0x010C and 0x010D – AUXPCM\_SEC\_AFE
  - 0x13B1 Ver1 – Audio vocoder Rx PCM; multiple Rx PCM files  
 \*.pcm.<sessionId>.<ChannelInterleave>.<dir>.wav  
 <ChannelInterleave> mapping:
    - 0x0102 – VPTX\_ECNEAR\_IN
    - 0x0202 – VPTX\_OUT
    - 0x0302 – VPTX\_ECFAR\_IN
    - 0x0402 – VPRX\_OUT
    - 0x0702 – VPTX\_INT\_OUT

- 0x13B0 Ver0 – Audio vocoder data paths; mingle Tx PCM file  
\*.pcm.tx.wav
- 0x13B1 Ver0 – Audio vocoder Rx PCM; single Rx PCM file  
\*.pcm.rx.wav
- ADSP logs – Rx/Tx PCM and bitstream files  
 <log code>.pcm.<sessionId>.<tapId>.<channel number>.<dir>.wav  
 <log code>.bitstream.<sessionId>.<tapId>.<media format Id>.<dir>.bin  
 <tapId> mapping for 0x1586:
  - 0x3100 and 0x3101 – INTERNAL\_BT\_SCO
  - 0x3102 – INTERNAL\_BT\_A2DP
  - 0x3104 and 0x3105 – INTERNAL\_FM
  - 0x4100 and 0x4101 – SLIMbus\_0
  - 0x4102 and 0x4103 – SLIMbus\_1
  - 0x4104 and 0x4105 – SLIMbus\_2
  - 0x4106 and 0x4107 – SLIMbus\_3
  - 0x4108 and 0x4109 – SLIMbus\_4
 <tapId> mapping for 0x158A:
  - 0x10F60 – VPTX\_ECNEAR\_IN
  - 0x10F61 – VPTX\_ECFAR\_IN
  - 0x10F62 – VPTX\_OUT
  - 0x10F63 – VPRX\_OUT
 <tapId> mapping for 0x158B:
  - 0x10F64 – VENC\_IN
  - 0x10F65 – VDEC\_OUT

**NOTE:** \*.labels.txt is also generated for the Audacity application (Audacity label file format). <dir> is either tx or rx.

■ Generated files from vocoder packets logged:

- Temporary frame file (.pkt) and PCM audio chunk file (.wav)
  - General format:  
 <file>.<ext>.voc.<dir>.frame.<#>\_logID\_<codec>.<ext>
  - 0x1804 / 0x1805 format:  
 <file>.<ext>.voc.<dir>.frame.<#>\_[[1<sup>st</sup> pkt  
 timestamp]]\_logID\_Instance\_Call#\_<codec>.<ext>
  - 0x1914 / 0x1915 format:  
 <file>.<ext>.voc.<dir>.frame.<#>\_[[1<sup>st</sup> pkt  
 timestamp]]\_logID\_VSID\_Call#\_VocoderSession#\_<codec>.<ext>



- Combined PCM file (.raw and .wav)
  - General format:  
`<file>.<ext>.voc.<dir>_logID_SampleRate.<ext>`
  - 0x1914 / 0x1915 format:  
`<file>.<ext>.voc.<dir>_logID_VSID_Call#_SampleRate.<ext>`  
`<file>.<ext>.voc.<dir>_logID_VSID_SampleRate.<ext>`

### 3.6.20.7 Default directory for generated files

The default directory where all generated files can be found is set to:

- C:\Users\userId\Vocoder for Windows
- /Users/userId/Public/Vocoder for Mac
- /usr2/userId/Public/Vocoder for Linux.

This directory is not cleaned up after you close QCAT.

### 3.6.20.8 Saving frame files

The Save Frames button for the vocoder streams allows the user to save the vocoder frame files that were sent to the C-SIM for decoding (\*.pkt) and the raw PCM that came from that frame file (\*.wav). The PCM format of the .wav file depends on the codec used. If multiple codecs were used in the same file, the .pkt files are named in sequential order and tagged with the vocoder used for that section.

## 3.6.21 Perl sample scripts

Scripts can be found at Qualcomm\QCAT 6.x\Script. ExtractPcmFiles.pl and ExtractVocoderData.pl process PCM packet log codes, e.g., 0x13B0, etc., vocoder packet log codes, e.g., 0x7143, etc., and get \*.pcm.\*.tx(rx).wav files and \*.voc.tx(rx).wav files.

### Usage

```
perl ExtractPcmFiles.pl -in <file> -out <dir> -codec <codec>
perl ExtractVocoderData.pl -in <file> -out <dir> -codec <codec>
```

Where:

- -in – The input log file (full path), either .isf, .dlf, or .qmdl
- -out – The output directory (full path); directory will be created if it does not exist
- -codec – The vocoder used
  - auto – Determine vocoder from log packets (0x7143, 0x7144, 0x14D0, 0x14D2, 0x1804, 0x1805, 0x1914, and 0x1915 only)
  - celp13k – Code Excited Linear Prediction 13 kbps
  - evrc – Enhanced variable-rate codec
  - evrc-b – Enhanced variable-rate codec B
  - evrc-wb – Enhanced variable-rate codec wideband
  - hr – GSM half-rate

- fr – GSM full-rate
- efr – GSM enhanced full rate
- amr-nb – Adaptive multirate (narrowband)
- amr-wb – Adaptive multirate (wideband); a Perl script that can be run from command line
- evrc-nw – Enhanced variable rate CODEC, SO 73
- eamr – Enhanced adaptive multi-rate
- evrc-nw2k – Enhanced variable rate CODEC, SO 77
- evs – Enhanced voice service
- g711 – G711 service

### 3.6.22 Call flow analysis support

A call flow page is shown here.

Time	Msg#/Ch	CM	CB	GSDI	GHDI	DS	SNDC
00:42:50.223	5						
00:42:51.019	6						CM_SIM_NOT_AWA
00:42:51.020	7						
00:42:51.020	8						
00:42:51.020	9						
00:42:51.021	10						
00:42:51.021	11						MS_CM_REGP I
00:42:51.138	12						CM_SERVIC
00:42:51.139	13						
00:42:51.140	14						
00:42:51.413	15						CM_ABORT_CC_REQ
00:42:51.413	16						CM_ABORT_CC_CONF
00:42:51.414	17						CM_ABORT_SS_REQ
	18						

Several components to this page are detailed below. There are also several ways to customize the output, explained in Section 3.6.22.5.

#### 3.6.22.1 Column information (top)

Columns across the top of the call flow page are shown here. The columns indicate the names of the message sender and recipient.

Time	Msg#/Ch	CM	CB	GSDI	GHDI	DS	SNDC
------	---------	----	----	------	------	----	------

**NOTE:** See Section A.2 for a current list of the column abbreviations.

### 3.6.22.2 Message index information (left)

The message index information across the left displays:

- Timestamp of the message delivered
- Message index number (this is the Nth CFA packet in the file)
- Channel information (optional) – When the message contains information received or Delivered Over-the-Air (OTA), the channel used is conveyed; channels would be one of:
  - BCCH – Broadcast channel
  - RACH – Random access channel
  - CCCH – Control channel (paging, access grant, etc.)
  - SACCH – Standalone control channel
  - DCCH – Dedicated control channel (either FACCH or SDCCH)

Time	Msg#/Ch
22:27:24.925	101 RACH
22:27:28.770	102 BCCH

### 3.6.22.3 Message summary information (center)

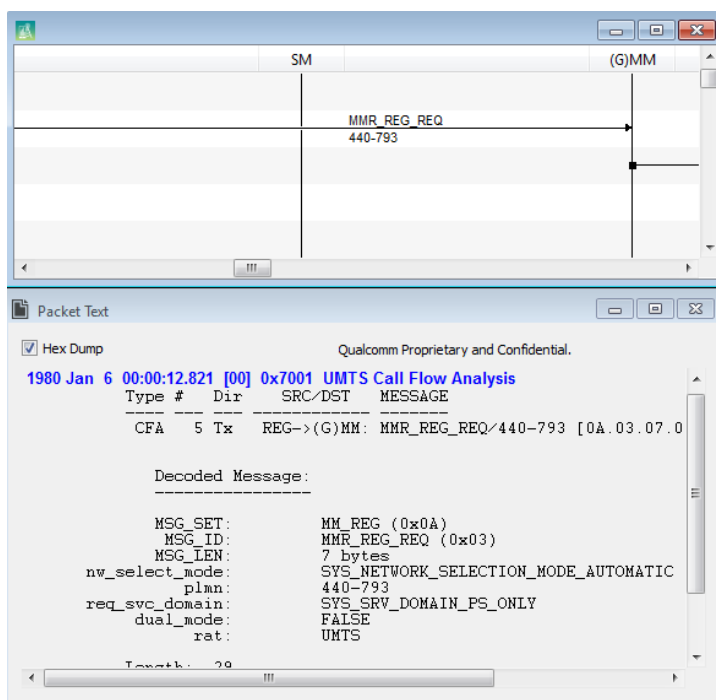
The message summary information displays the name of each message (above the arrow) and its direction (the arrow head). Messages sometimes have additional information (below the arrow), e.g., if the message contains a status field, the success or failure may be displayed as the subtext. In addition, color may be used to emphasize the message. By convention:

- **GREEN** is used when the messages have successful results.
- **RED** is used when messages contain failure information.
- **GRAY** is used to deemphasize the subtext, e.g., Over-the-Air (OTA) message names are displayed as subtext for the DATA\_IND conduit message; for messages that happen frequently, such as the system information messages, the subtext is gray.
- **BLUE** is used to emphasize the subtext, e.g., *PAGING REQUEST TYPE* is normally gray if there is no paging information contained in it but blue if there is information. This allows you to quickly find the page request of choice.

The message detail information displays fields for the particular message. To access this information, click the hyperlink (the message name above the arrow) of the message in which you are interested. This displays:

- MSGHEX – Raw payload of the message
- MSG – Fields and values of the message (if the message is known by the parser)

- Silk L3 – OTA decode (if the message contains OTA message content)



### 3.6.22.4 CFA text files

The following is an example of a single CFA entry in the ASCII text file:

```

-----
Time   Channel Type   # Dir   SRC/DST   MESSAGE
-----
00:01:44.060   DCCH   CFA 00104   Rx (G)MM<- (G)RR   RR_DATA_REQ
[CC_CONNECT_ACK]

                                           [09.01.02.00.03.4F]

Decoded Message:
-----
MSG_SET:      MM_RR (0x09)
MSG_ID:      RR_DATA_REQ (0x01)
MSG_LEN:      2 bytes
layer3_message: (2 bytes)
03.4f.

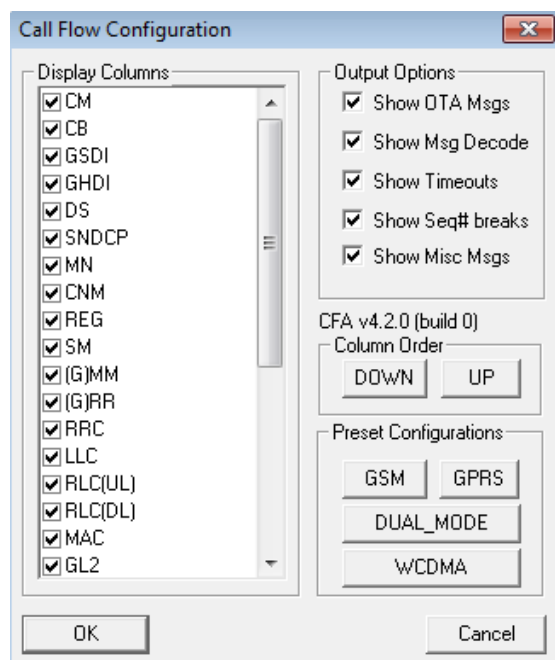
SILK (L3 Message):
-----
chan_type = 0
trans_id_or_skip_ind = 0
prot_disc = 3 (GSM_CALL_CONTROL)
msg_type = 15
prot
  call_ctrl_prot
    CONNECT_ACKNOWLEDGE
  
```

It contains the same information available in the standard call flow pages, including:

- Timestamp of the CFA message
- Channel name (optional)
- Message index number (this is the Nth CFA packet in the file)
- Message direction (Tx/Rx)
- Message sender and receiver (SRC/DST respectively)
- Message name
- Submessage name
- Hexadecimal representation
- Decoded message content
- Silk L3 – OTA decode (if the message contains OTA message content)

### 3.6.22.5 Configuring CFA

There are several configuration options that are stored in the CFA message parser library itself. You can reach this by selecting **View CFA Configuration** from the View menu. If you click **OK**, these settings will be written to the registry and applied the next time you parse a file. In other words, it does not take effect on the current file(s); you will have to close and reopen them.

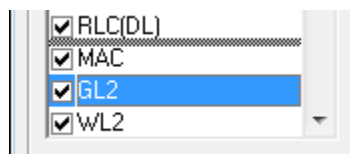


### 3.6.22.5.1 Column selection and order

One of the powerful features of CFA is the ability to display, hide, and reorder the columns. This allows you to produce output in a way that is familiar (such as matching a documented call flow). There are several ways to accomplish this; the recommended order for quickly achieving the desired column output is:

1. Select the desired column order.
  - Clicking **DOWN** enables all columns and places them in their natural descending order (highest software entity on the left; lowest on the right). This is the default setting.
  - Clicking **UP** enables all columns and places them in their natural ascending order.
2. Select the desired preset configuration.
  - Clicking **GSM** displays the columns CM, MN, CNM, REG, MM, RR, GL2, and GSM/GPRS Layer 1 (GL1).
  - Clicking **GPRS** displays the columns CM, SMDCP, MN, CNM, REG, SM, GMM, GRR, LLC, RLC(UL), RLC(DL), MAC, GL2, and GL1.
  - Clicking **WCDMA** displays the columns CM, MN, CNM, REG, MM, RRC, RLC(UL), RLC(DL), MAC, WL2, and WL1.
  - Clicking **DUAL\_MODE** displays the union of GSM and WCDMA options.
3. Turn individual columns on or off by checking or unchecking respectively the box next to the column name.

You can also reorder individual columns by clicking the column name and dragging it to its desired location. An example of dragging the GL1 column up under the GL2 column is shown here.

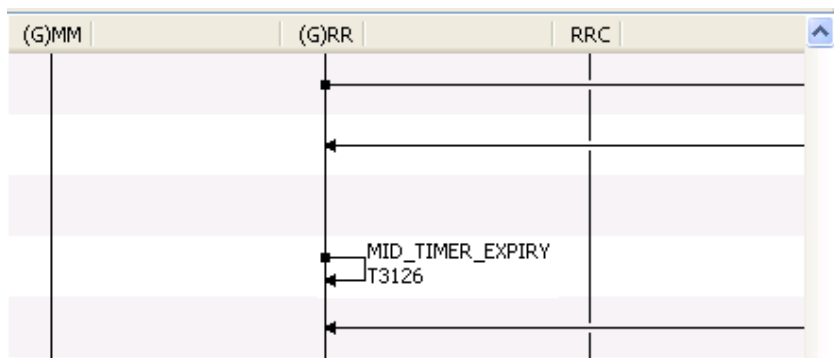


**NOTE:** Messages that have either their destination or source column hidden are either placed on the MISC column or hidden.

### 3.6.22.6 Timeouts

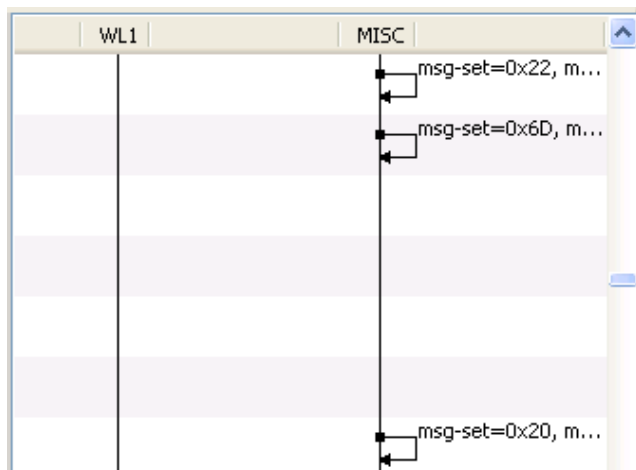
Timer expirations are a special class of messages that are self-posted to the column they are intended to notify. Because of this, they are difficult to selectively hide, e.g., if the RR column is checked, any RR timer expiration would be visible. To address this, the **Show Timeouts** checkbox can be unchecked to hide these messages, if desired.

An example timeout, displayed if **Show Timeouts** is checked, is shown here.



### 3.6.22.7 Miscellaneous column

Messages that have been hidden are lumped on the MISC column heading at the far right unless you uncheck this option. An example timeout, displayed if **Show Timeouts** is not selected, is shown here.



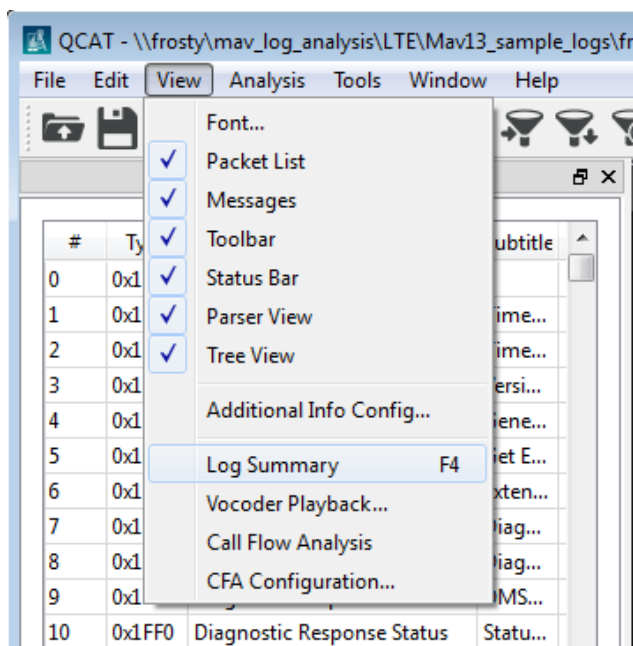
### 3.6.22.8 Sequence breaks

Each CFA packet contains a sequence number stamped and incremented by the phone. This mechanism is to help determine if the entire call flow was present or if there were packets dropped due to performance reasons (or other problems). The message in the following screenshot is visible in the call flow if **Show Seq# breaks** is checked.

Msg#/Ch	CM	CB
189		
***Missing 67 Messages***		
1		
2		

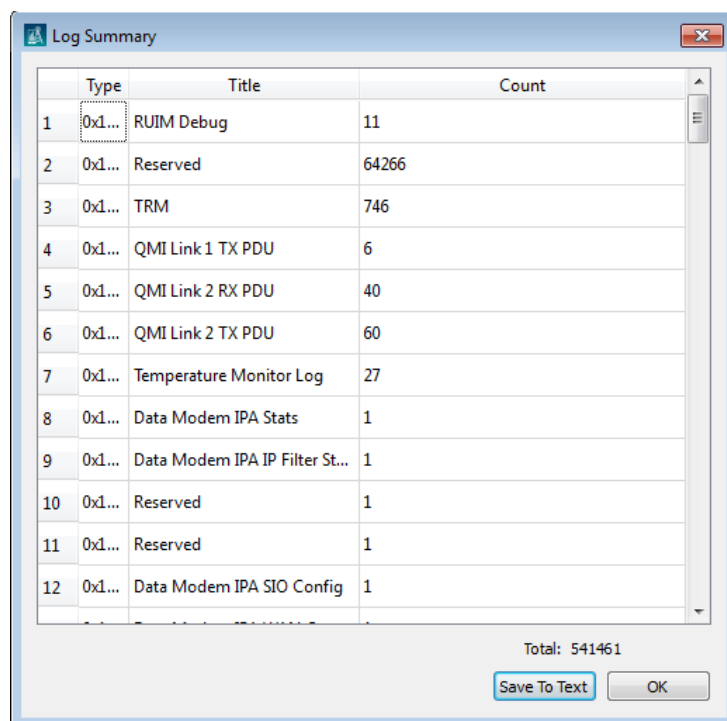
### 3.6.23 Viewing the log summary

The Log Summary display groups the log packets by type and shows the count of each packet type in the currently opened log file. It also displays the total count of all log packets.





To invoke the log summary, select View→Log Summary. The shortcut for this function is F4.



The Log Summary dialog box displays a table with the following data:

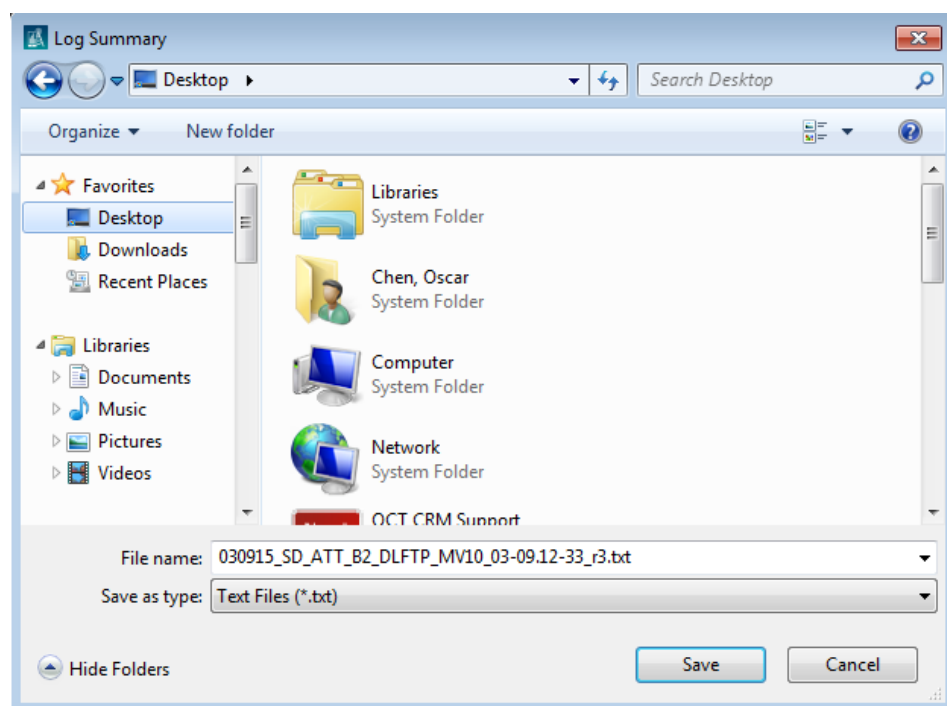
	Type	Title	Count
1	0x1...	RUIIM Debug	11
2	0x1...	Reserved	64266
3	0x1...	TRM	746
4	0x1...	QMI Link 1 TX PDU	6
5	0x1...	QMI Link 2 RX PDU	40
6	0x1...	QMI Link 2 TX PDU	60
7	0x1...	Temperature Monitor Log	27
8	0x1...	Data Modem IPA Stats	1
9	0x1...	Data Modem IPA IP Filter St...	1
10	0x1...	Reserved	1
11	0x1...	Reserved	1
12	0x1...	Data Modem IPA SIO Config	1

Total: 541461

Buttons: Save To Text, OK

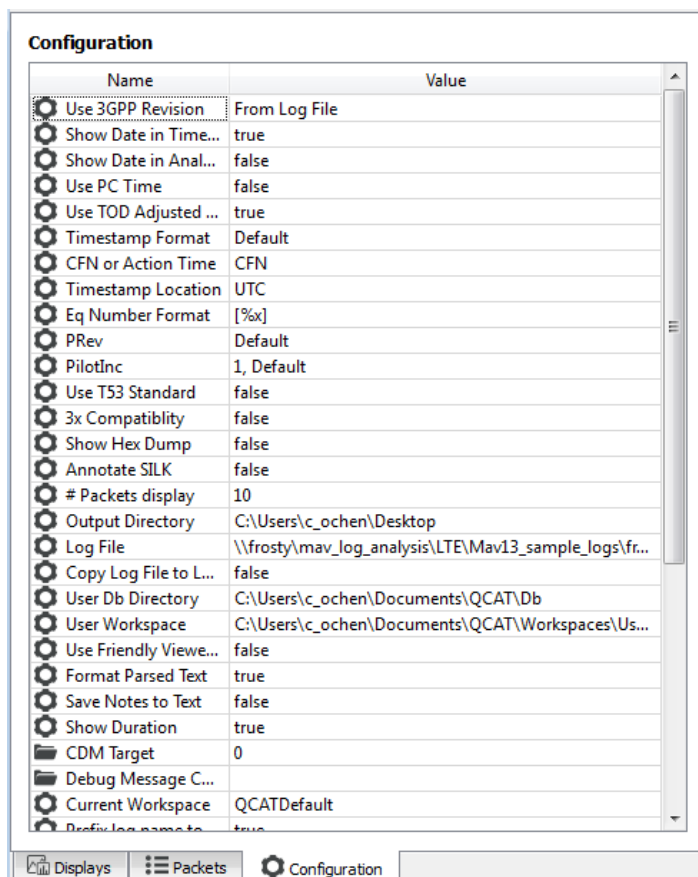
### 3.6.24 Saving the log summary

To save the log summary information to a .txt file, click **Save to Text** in the Log Summary dialog. This brings up the Windows Save As dialog where you select the directory and filename for the summary information.



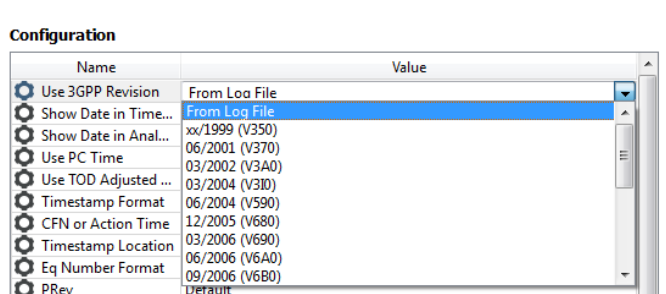
### 3.6.25 Configuration tab

The Configuration tab replaces the Option menu found in previous versions. It contains a superset of those features.



#### 3.6.25.1 Use 3GPP revision

QCAT will normally use the 3GPP revision number found in the log file. It is possible that the number in the file is incorrect. This feature allows the user to override the From Log File value and specify the 3GPP revision number to use.



### 3.6.25.2 Show date in Time Column

This option can be set to True or False. When this option is enabled, QCAT shows the date and time in the time column (YYYY Mon DD HH:MM:SS.sss). When it is disabled, only the time (HH:MM:SS.sss) is displayed.

### 3.6.25.3 Show date in Debug Message file

This option can be set to True or False. When this option is set to True, QCAT includes the date with the timestamp when writing out a debug file. When it is set to False, it writes out only the timestamp. This can be seen by selecting File→Save→Text..., checking **Save Debug Messages Only** and clicking **Save**.

### 3.6.25.4 Use PC time

When this option is enabled (True), QCAT offsets all timestamps by PC Time offset. The value is a diff between PC Time from Diagnostic Version Packet and first Nonzero Timestamp in the log. The first Nonzero Timestamp should not be further than 1 hr from the logged PC Time in the Diagnostic Version Packet. This way QCAT will ignore stale packets that sometimes are present in the log.

To access this option, click the **Configuration** tab. To toggle this option to True or False, single-click the **Use PC Time** option to activate the drop-down list. Select the desired value. Changes take effect immediately. The shortcut for this function is Alt+C.

**NOTE:** This option only works for specific DM versions.

### 3.6.25.5 Timestamp format

The Timestamp Format option has three settings, i.e., Default, Calendar, and Days. This option affects the way the date is displayed. The Days format is (Day DD HH:MM:SS.sss) while the Calendar format is (YYYY Mon DD HH:MM:SS.sss). The default for UMTS targets is Day. The default for CDMA targets is Calendar.

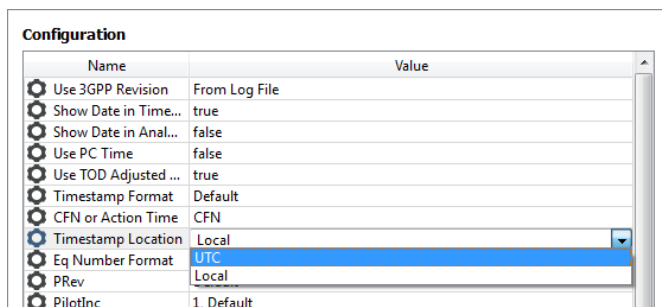
**NOTE:** If Use PC Time is set to True, the setting of the Timestamp Format option is ignored. It will always be the Calendar format.

#### Configuration

Name	Value
Use 3GPP Revision	From Log File
Show Date in Time...	true
Show Date in Anal...	false
Use PC Time	false
Use TOD Adjusted ...	true
Timestamp Format	Default
CFN or Action Time	Default
Timestamp Location	Calendar
Eq Number Format	Days

### 3.6.25.6 Timestamp location (UTC vs. local time)

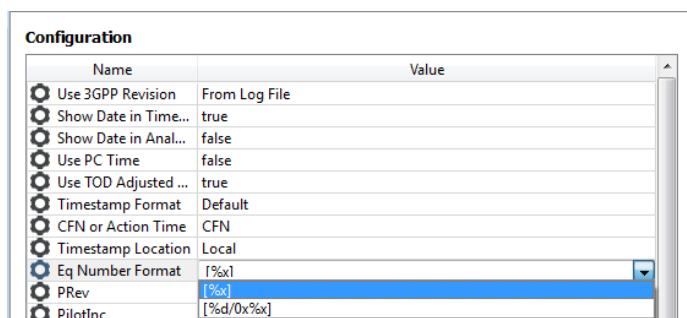
The Timestamp Location option allows users to show the timestamp in the local time zone instead of UTC time. This is simply an offset of as many hours as the local time is removed from UTC time.



### 3.6.25.7 Eq number format

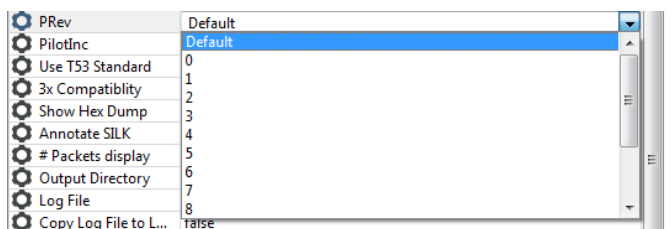
The Eq Number Format is a CFN number in the UMTS/WCDMA timestamps. For CDMA timestamps, it is an Action Time. The two options are either hex only [%x] or decimal and hex [%d/0x%x], e.g., if the packet's CFN is 0x5E, it can be displayed as one of the following:

- Day 2404 01:43:37.962 [94/0x5E] 0x4110 WCDMA Active Set
- Day 2404 01:43:37.962 [5E] 0x4110 WCDMA Active Set



### 3.6.25.8 PRev

A PRev revision is needed to decode CDMA messages. This allows you to override the default and set the correct revision number.



### 3.6.25.9 Use T53 standard

The Use T53 Standard option can be set to True or False. To toggle this option on and off, select Options→Use T53 Standard. The shortcut for this function is Alt+5. This option applies to CDMA targets only. When this option is enabled, QCAT notifies the signaling message parsing routines to interpret the messages as defined in the ARIB T53 Standard, as opposed to the IS-95A Standard.

### 3.6.25.10 3x compatibility

The 3x Compatibility option can be set to True or False. When True, it allows QCAT to work with packets created for QCAT Ver 3x.

### 3.6.25.11 Show hex dump

The Show Hex Dump option can be set to True or False. Toggling this option has the same effect as toggling the Hex Dump checkbox in the parser view.

### 3.6.25.12 Annotate SILK

Enabling the Annotate SILK option will cause the OTA message parser (SILK) to annotate the text output with information about the raw bits that were decoded into a particular output field.

```

Hex Dump
2004 Apr 27 05:17:23.571 [1C] 0x1008 Forward Channel Traffic Message -- Extended S Channel Assignment Msg
0000 004 0110..... protocol_rev = 6 (IS2000 Rev 0)
0004 004 0011..... chan_type = 3 (Forward Traffic)
0008 000
0008 008 00010100..... record_len = 20
0016 000 (begin_record(pre=8,post=16): record length computed = 136 bits)
0016 000 ftc_msg
0016 008 00100011..... gen
0024 000 msg_type = 35 (Extended Supplemental Channel Assignment)
0024 000 escam
0024 003 101..... hdr
0027 003 001..... ack_seq = 5
0030 001 0..... msg_seq = 1
0031 002 00..... ack_req = 0
0033 003 000..... encryption = 0
0036 004 0000..... start_time_unit = 0
0040 001 0..... rev_sch_dtx_duration = 0
0041 001 0..... use_t_add_abort = 0
0042 001 0..... use_sch_seq_num = 0
0043 001 0..... add_info_incl = 0
0044 002 00..... rev_cfg_included = 0
0046 001 1..... num_rev_sch = 0
0047 001 0000..... for_cfg_included = 1
0048 005 0000..... for_sch_fer_rep = 1
0053 000 0..... num_for_cfg_recs = 0
0054 004 0000..... for_cfgs[0] (99 bits left)
0058 004 0010..... for_sch_id = 0
0062 003 000..... sccl_index = 0
0065 000 for_sch_rate = 2
0065 009 000001000..... num_sup_sch = 0
0074 001 0..... sup_sch[0] (87 bits left)
0075 011 00000001001..... pilot_pn = 8 (0x8)
0086 002 00..... add_pilot_rec_incl = 0
0088 002 01..... for_sch_cc_index = 9 (0x9)
0090 000 qof_mask_id_sch = 0
0090 001 0..... num_for_sch = 1
0091 004 1111..... for_schs[0] (62 bits left)
0095 001 1..... for_sch_id = 0
0096 005 01000..... for_sch_duration = 15
0101 004 0000..... for_sch_start_time_incl = 1
0105 001 1..... for_sch_start_time = 8
0106 003 001..... sccl_index = 0
0109 001 0..... fpc_incl = 1
0110 001 0..... fpc_mode_sch = 1
0111 002 01..... fpc_sch_init_setpt_op = 0
0113 000 fpc_sch_chan = 0
0113 001 0..... num_sup = 1
0114 005 01010..... sups[0] (39 bits left)
0119 008 00011100..... sch_id = 0
0127 008 00010000..... fpc_sch_fer = 10
0135 008 01000000..... fpc_sch_init_setpt = 28
0143 001 0..... fpc_sch_min_setpt = 16
0144 001 0..... fpc_sch_max_setpt = 64
0145 007 0000000..... fpc_thresh_sch_incl = 0
0152 016 0100010111010100 rpc_incl = 0
(end_record consumes 7 bits: record started at 16)
reserved 16 bits = 0x45d4

Length: 32
Header: 20 00 08 10 C0 59 28 17 24 E1 8E 00
Payload: 14 23 A4 00 03 00 08 02 00 24 5F 40
48 94 38 20 80 00 45 D4

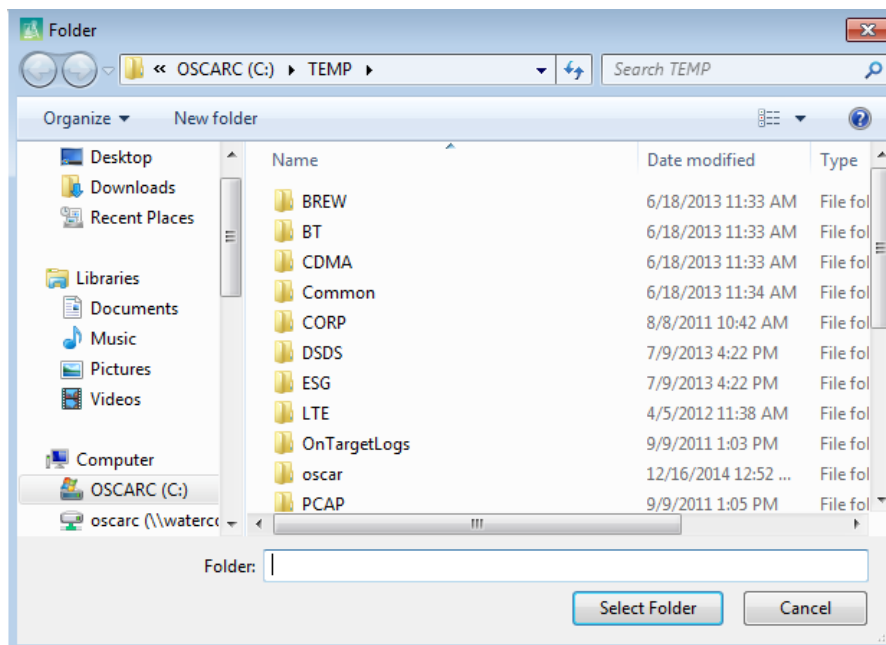
```

### 3.6.25.13 # Packets display

The # Packets display option specifies the maximum number of packets that are displayed at the same time in the right panel.

### 3.6.25.14 Output directory

The Output Directory option specifies the default output directory when exporting files.



### 3.6.25.15 Log file

The log file option shows the last log file loaded.

### 3.6.25.16 Copy log file to local machine

The Copy Log File to Local Machine option can be set to True or False, telling QCAT whether or not to copy log files that are accessed remotely to the local machine. The files are copied to the current user's %TEMP% directory.

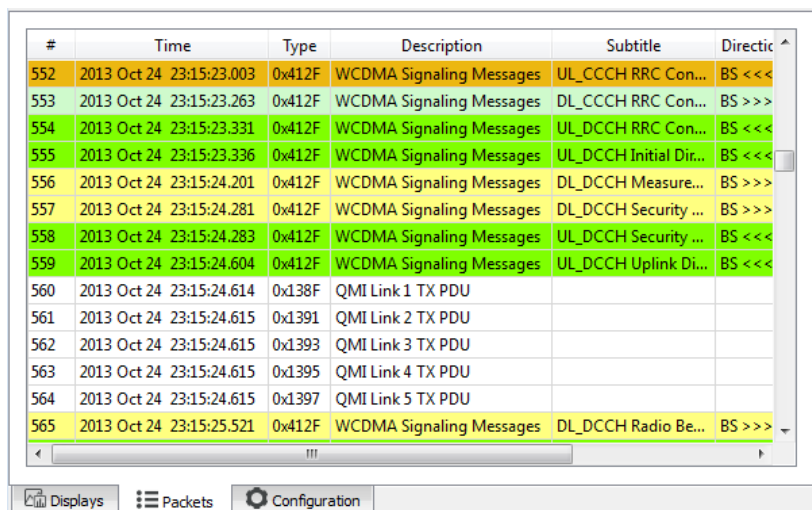
### 3.6.25.17 User Db directory

The User Db Directory specifies a directory in which user-defined diagnostic database files are located. These files will be loaded and given priority for parsing packets. Packets parsed by user-defined databases will be indicated in the supported log list by the name of the database file being used following the name of the log packet. The user database file will also be shown in the parsed output for that packet.

User parser libraries are also supported from the User Db directory. All user libraries that conform to the QXDM interface are loaded and used for parsing packets matching the registered IDs. For more information regarding these libraries, refer to the documentation and templates included with QXDM.

### 3.6.25.18 Use friendly viewer colors

The **Use Friendly Viewer Colors** option can be set to True or False, telling QCAT whether to use the Friendly Viewer application's color scheme when displaying packets in the left panel. When enabled, it color-codes 0x412F WCDMA signaling message packets based on their subtitle; otherwise, there is no color coding.



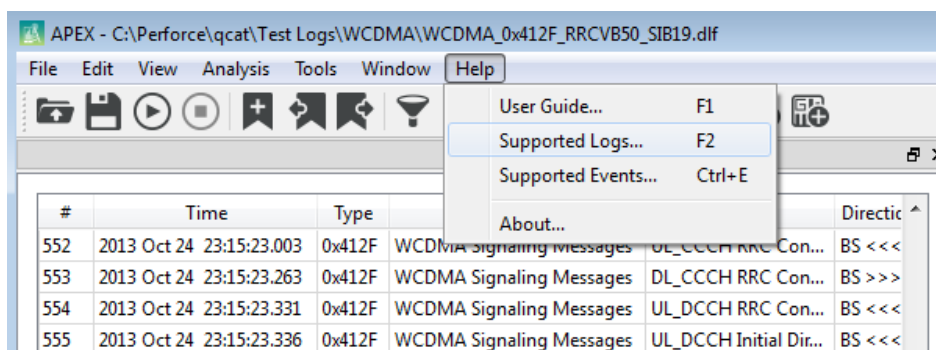
#	Time	Type	Description	Subtitle	Direction
552	2013 Oct 24 23:15:23.003	0x412F	WCDMA Signaling Messages	UL_CCCH RRC Con...	BS <<<
553	2013 Oct 24 23:15:23.263	0x412F	WCDMA Signaling Messages	DL_CCCH RRC Con...	BS >>>
554	2013 Oct 24 23:15:23.331	0x412F	WCDMA Signaling Messages	UL_DCCH RRC Con...	BS <<<
555	2013 Oct 24 23:15:23.336	0x412F	WCDMA Signaling Messages	UL_DCCH Initial Dir...	BS <<<
556	2013 Oct 24 23:15:24.201	0x412F	WCDMA Signaling Messages	DL_DCCH Measure...	BS >>>
557	2013 Oct 24 23:15:24.281	0x412F	WCDMA Signaling Messages	DL_DCCH Security ...	BS >>>
558	2013 Oct 24 23:15:24.283	0x412F	WCDMA Signaling Messages	UL_DCCH Security ...	BS <<<
559	2013 Oct 24 23:15:24.604	0x412F	WCDMA Signaling Messages	UL_DCCH Uplink Di...	BS <<<
560	2013 Oct 24 23:15:24.614	0x138F	QMI Link 1 TX PDU		
561	2013 Oct 24 23:15:24.615	0x1391	QMI Link 2 TX PDU		
562	2013 Oct 24 23:15:24.615	0x1393	QMI Link 3 TX PDU		
563	2013 Oct 24 23:15:24.615	0x1395	QMI Link 4 TX PDU		
564	2013 Oct 24 23:15:24.615	0x1397	QMI Link 5 TX PDU		
565	2013 Oct 24 23:15:25.521	0x412F	WCDMA Signaling Messages	DL_DCCH Radio Be...	BS >>>

### 3.6.26 Invoking the QCAT user guide

To invoke the QCAT user guide, select Help→User Guide. The shortcut for this function is F1. This invokes the PDF version of this user guide using the appropriate document viewer.

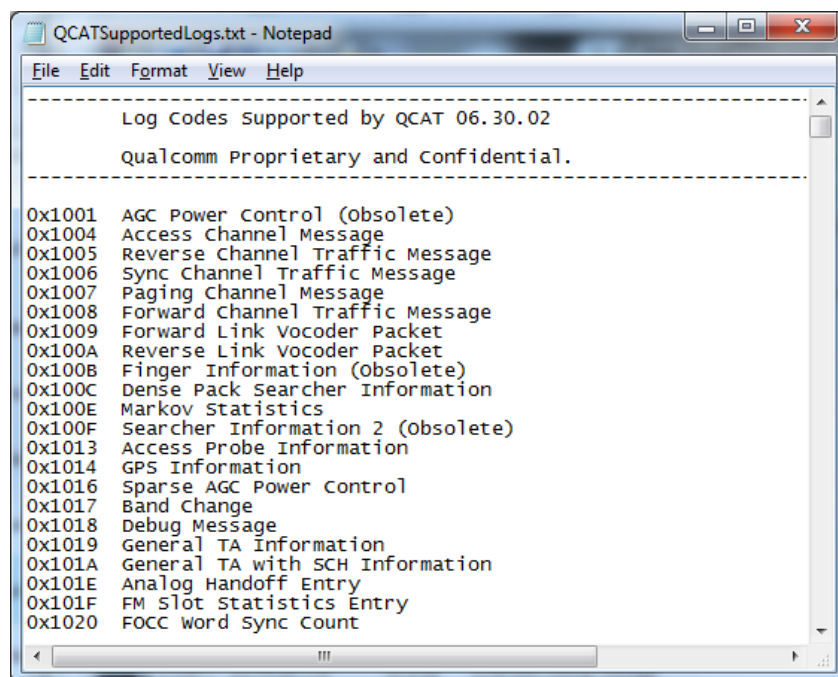
### 3.6.27 Viewing the Supported Logs list

QCAT creates a list of all log packets that it is capable of parsing by querying its internal parsing engine. This list is autogenerated each time you request this information. To view this list, select Help→Supported Logs. The shortcut for this function is F2.



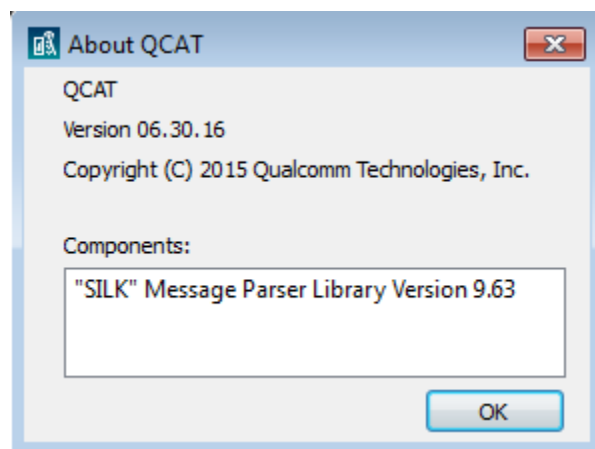


QCAT creates a .txt file with this information and invokes the Windows default text editor (usually Notepad) to display this .txt file.



### 3.6.28 Getting QCAT version information

The QCAT version and copyright information is found in the About dialog. To invoke the About dialog, select Help→About QCAT.



In addition to the QCAT product version, this dialog also displays the versions of other nonnative components such as the SILK message parser library.

### Exiting the application

To exit the application, select File→Exit.

## 3.6.29 Analysis features

These features are available when working with QCAT analyzers. To access the analyzers, select the Displays tab of the left view.

### 3.6.29.1 Importing workspace items

To add items to your workspace that have been saved from another instance of QCAT, select File→Import Workspace Items and choose the desired workspace item file (.awsi). Any item definitions in this file will be added to the user workspace.

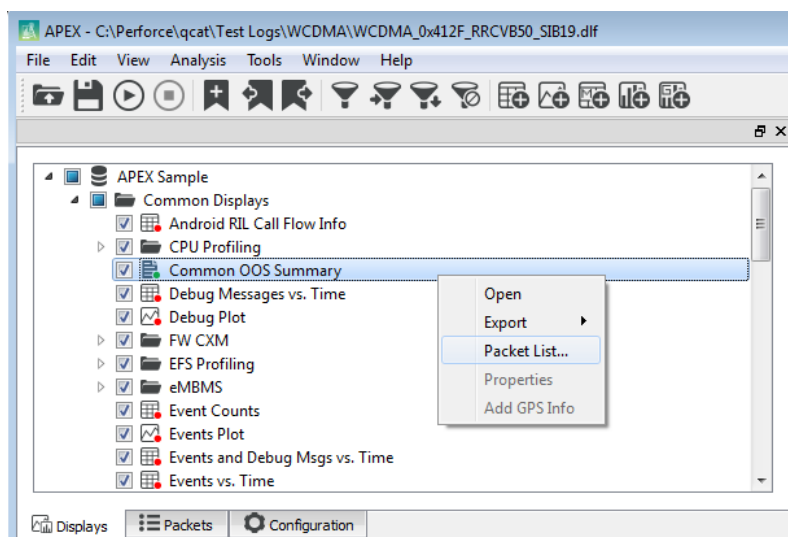
Workspace item files (.awsi) are generated by selecting items in the workspace view, right-clicking, and selecting Export item. This is designed to allow collaboration between team members by sharing definitions of specific items.

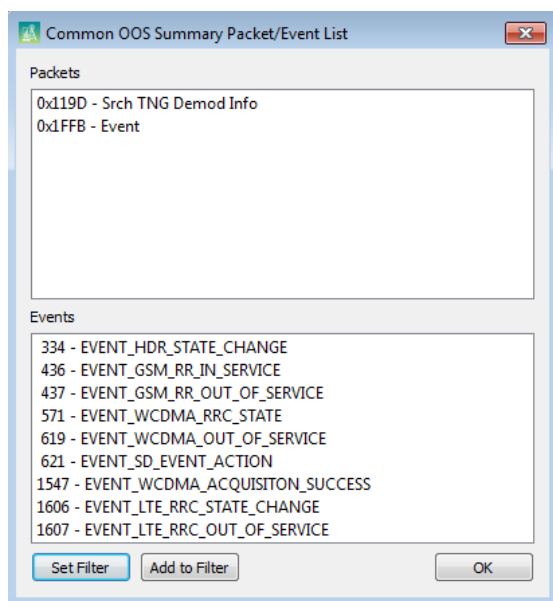
### 3.6.29.2 Viewing an analyzer

To process and view any analyzer listed in the left view, double-click the analyzer name, or select the analyzer and then select Display→Open from the menu bar. The analyzer is displayed in the right view.

### 3.6.29.3 Viewing the packet and event types used by an analyzer

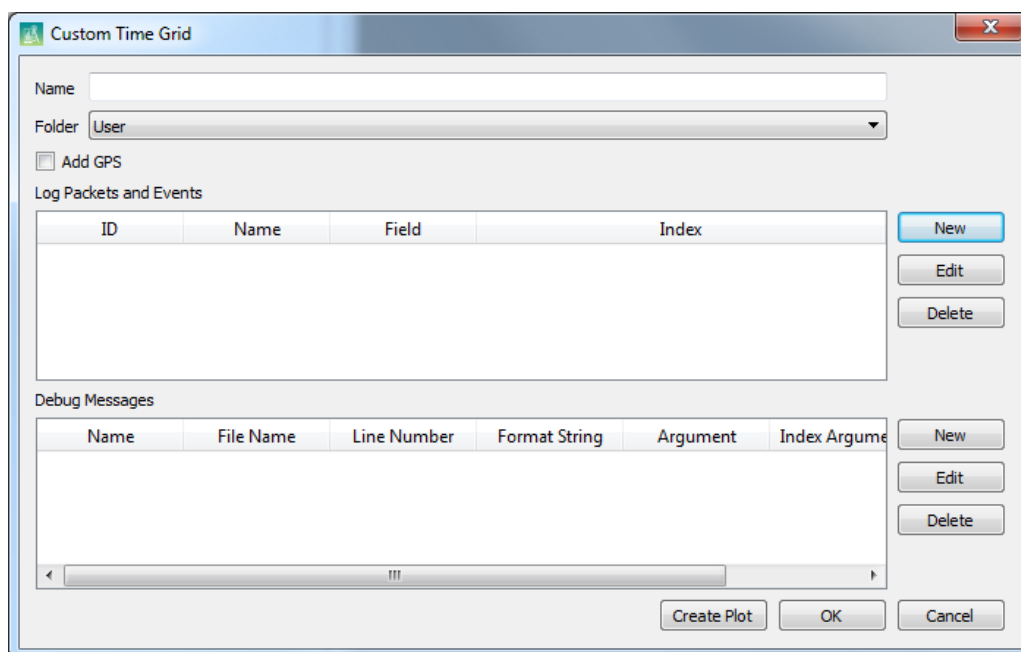
To view the list of log packets and events that an analyzer uses, right-click the analyzer name and select Packets list, or select the analyzer and then select Display→Packets list from the menu bar; the dialog box displays.





### 3.6.29.4 Creating a custom grid display

QCAT allows the creation of time grids for most of the modern packet fields (excluding OTA packets) and debug messages. To create a new time grid, select **Analysis > New Grid**. This brings up the custom time grid window shown below.



Each row in this dialog corresponds to a column in the resulting time grid. The dialog fields are:

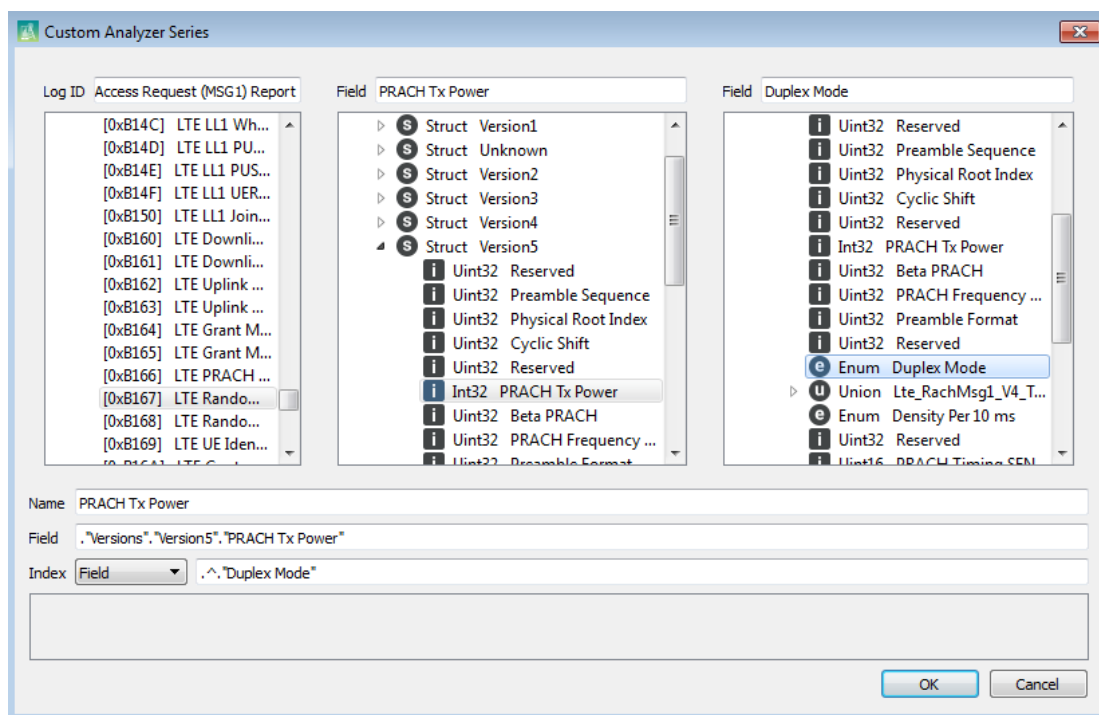
- Name – Name of the new analyzer
- Folder – Workspace folder in which to insert the analyzer
- Add GPS – Inserts interpolated latitude and longitude values into the grid for each entry

The following is present in each grid based on a log packet:

- Type – What type of packet used (Log, Event, Debug Message)
- ID – ID of the packet/event/debug message to use
- Name – Name of the column in the grid
- Field – Accessor used internally to access the field
- Index – Accessor to an indexing field

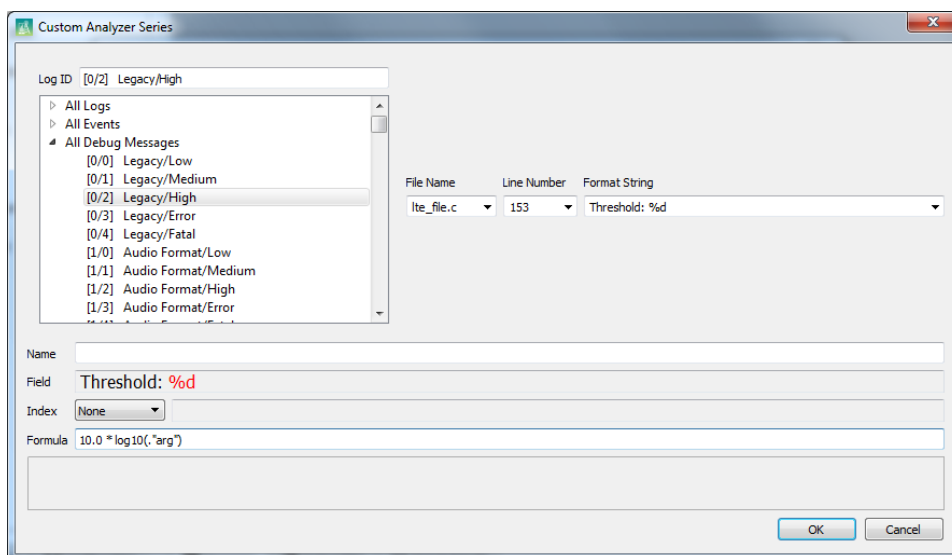
To make a plot from this grid, click **Create Plot**, which creates a default plot using every column. To customize the resulting plot, see Section 3.6.29.6.

Clicking **New** or **Edit** will bring up the Custom Plot Series dialog to select a field to be plotted.



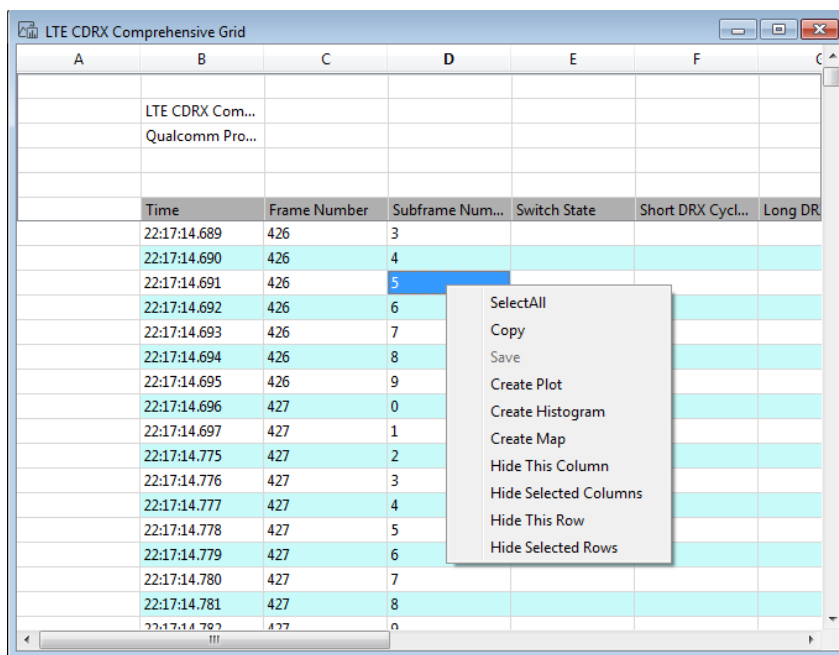
Click the log ID and field desired. Optionally, you can select an index. The index works as follows, for every unique value in the index field, a new column will be made for the field, e.g., this is useful for making a time grid of Tx power, where the Tx power is broken up per PSC. Therefore, the field would be Tx power, the index would be PSC. In the screenshot example above, it would create a PRACH Tx power (duplex mode = FDD) column and a PRACH Tx power (duplex mode = TDD) column in the final output. If the field is in an array, array index can also be selected. So, if the field is in an array of five elements, five columns are created for Field[0] to Field[4].

If a debug message is selected and a log file is loaded, the file name and format string combo boxes are filled with the strings available in the log file. Selecting one of them automatically populates the possible options for the other combo boxes. Once a debug message is selected (file name, line number, and format string) then the argument to be used must be selected from the format string. Clicking on the argument in the format string selects that argument as the data source for the series. Checking the Index check box allows a separate argument to be selected for indexing the first argument. The screenshot below shows how this works. An optional formula can also be applied to the value from the debug message by modifying the Formula field. Any valid C operations can be used, along with log10, pow, sin, cos, and tan.



### 3.6.29.5 Customizing a grid display

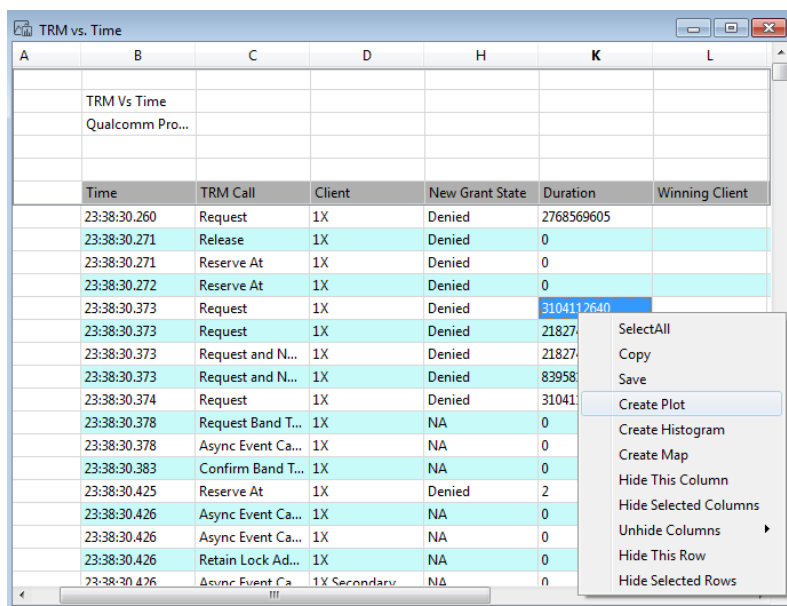
An existing grid display can be customized by hiding columns, as shown here. To hide a column, right-click in any cell in the column to be hidden and select **Hide This Column**. To hide a group of columns, use **Shift+Click** to select adjacent columns or **Ctrl+Click** to select individual columns, then right-click and select **Hide Selected Columns**.



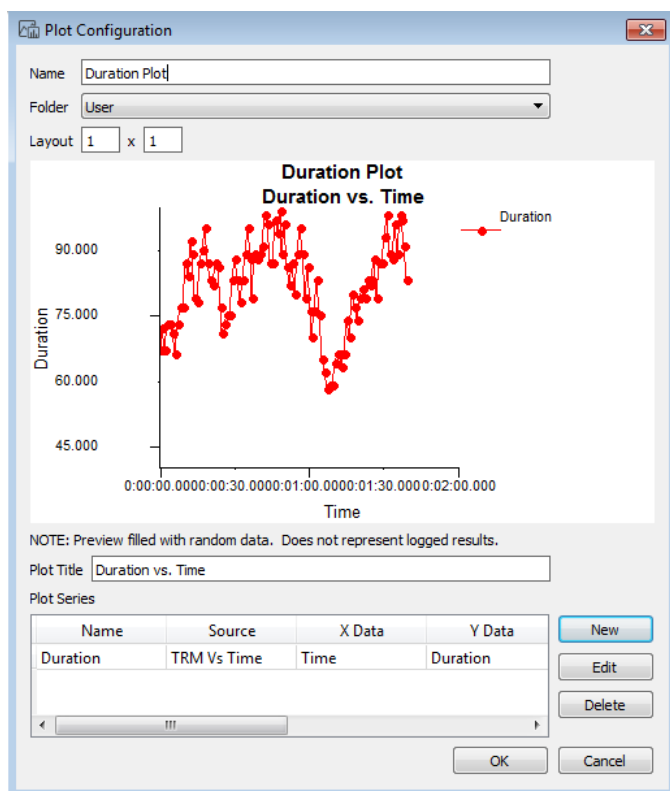
When a custom grid is closed, QCAT asks if the display configuration should be saved to the user displays section of the workspace. Saving the display allows you to open the grid in the same configuration at a later time.

### 3.6.29.6 Creating a custom plot display

A custom plot display can be created from any grid file by right-clicking any cell in the column to be plotted and selecting Create Plot.



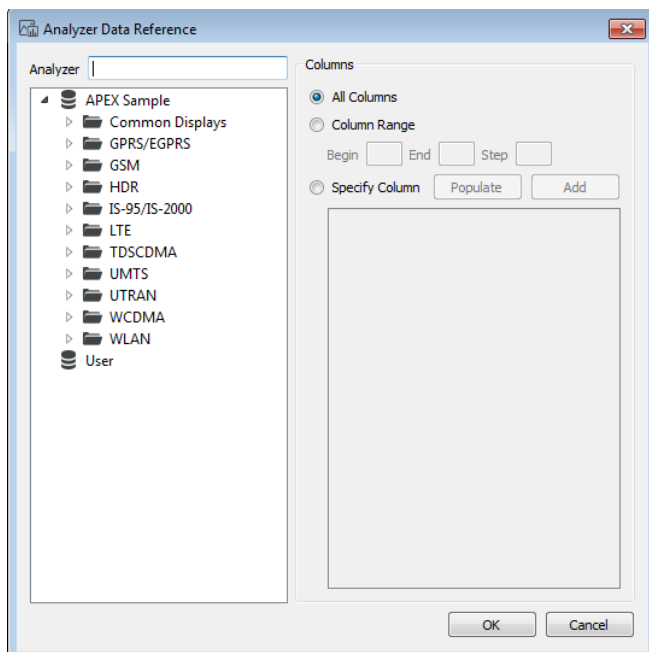
When Create Plot is selected, the Edit Figure dialog appears.



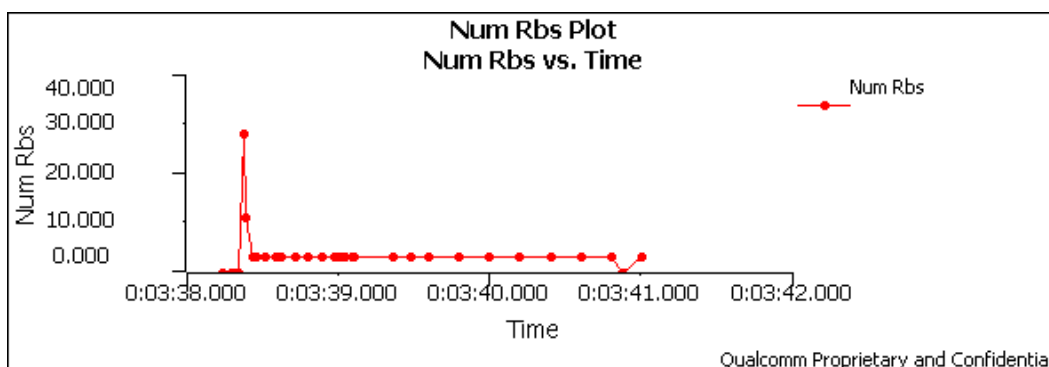
The title of the figure is displayed in the title bar of the window containing the figure. It will also be the label of the node in the user displays section of the workspace.

### 3.6.29.6.1 Adding and customizing subplots

A figure may contain multiple subplots but often contains only one. The subplot created by right-clicking the column in the grid view will by default be a scatter plot that uses the first column in the grid as the x-axis and the selected column as the y-axis. To change the parameters of the subplot, select a subplot from the list on the Edit Figure dialog and click **Edit**. To add a new subplot, click **New**. To delete the selected subplot, click **Delete**. When New or Edit is selected, the Subplot Info dialog appears.



The subplot title is displayed at the top of the subplot. The layout allows customization of the number and shape of subplots on a figure. The figure is divided into a grid with columns (Cols) and rows (Rows). The num field specifies which cell of the grid this subplot will occupy (numbered from 0, left-to-right and top-to-bottom). When layout is set to 1,1,0, the subplot fills the entire figure. However, to put multiple subplots on a figure, each subplot rectangle can be customized. A sample custom subplot with layout set to 2,1,1 is shown.

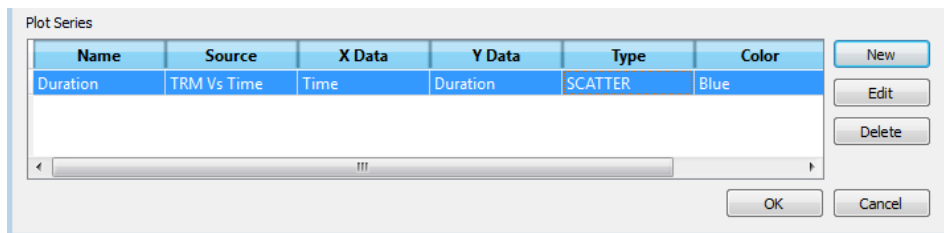


The primary X label field is the label for the x-axis on the bottom of the subplot. The primary Y label field will be the label for the y-axis along the left side of the subplot. The secondary x-axis is along the top of the plot and the secondary y-axis is along the right side.



### 3.6.29.6.2 Adding and customizing a series

To add another series to a subplot, click **New** on the Subplot Info dialog. To modify an existing series, select the series in the list and click **Edit**. When New or Edit is selected, the Series Info dialog appears.

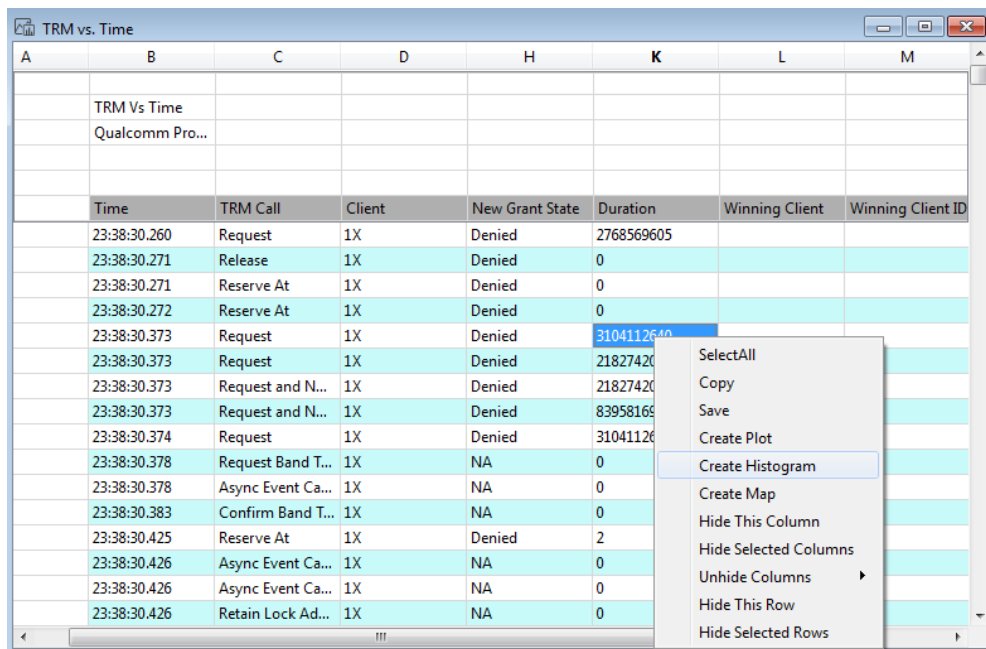


The fields in this dialog are:

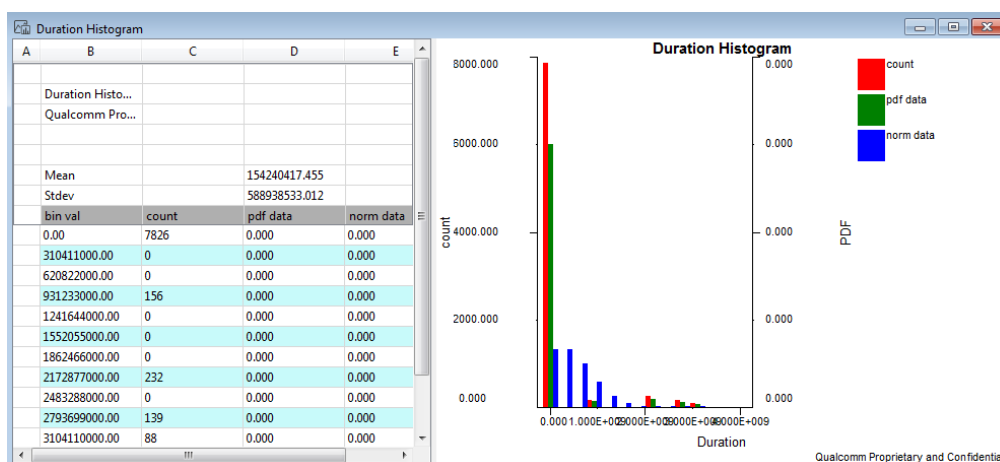
- **Name** – This specifies the name that the series will be given in the legend.
- **Analyzer Table** – This specifies the analyzer that contains the data to be plotted.
- **X Data** – This specifies the table column that contains the (independent variable) x-axis values; the column can also be specified using the number of the column (0-based).
- **Y Data** – This specifies the column that contains the (dependent variable) y-axis values. Although usually one series is tied to one column, a series can be tied to a range of columns. This is often used for tables where the number of columns is not known in advance. To specify a range of columns for the y data, click the combo drop-down list and select <Range...>.
- **Type** – This specifies how to draw the series (line, scatter, vertical bar, etc.).
- **X Axis Type** – This specifies whether the x-axis series contains values that can be compared and put in order (value) or if the series merely contains labels for the x-axis (category). If the x-axis is a time value, the axis is usually treated as a value axis so that time values appear in order along the x-axis.
- **Y Axis** – This specifies whether the y-values are scaled along the primary or secondary y-axis.

### 3.6.29.7 Creating a histogram from a grid column

QCAT allows a histogram to be created from any column of data in an existing grid. Right-click the column and select Create Histogram. This causes a new histogram to be added to the user workspace section.

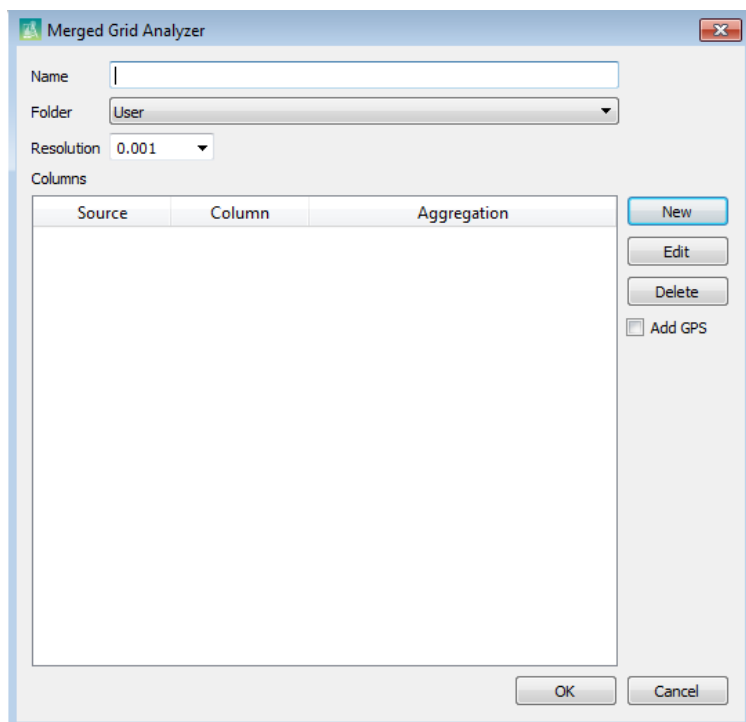


The resulting histogram can then be exported to text, Excel, or the screen just like any other analyzer. An example of the output is shown here; it will contain the histogram of value counts, CDF, and PDF.



### 3.6.29.8 Creating a custom merged grid

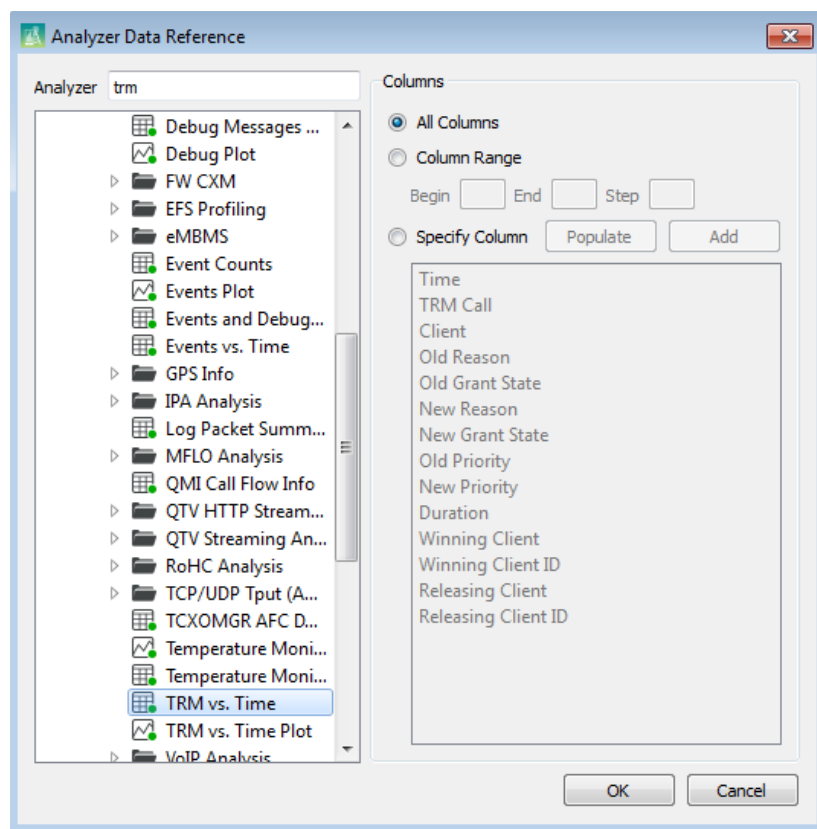
You can create a new grid analyzer based on the fields of two existing analyzers. To create a merged grid, select Analysis→New Merged Grid or right-click User→New Merged Grid.



The fields in this dialog are:

- Name – The name of the new merged grid analyzer
- Folder – The output folder; defaults to user workspace
- Resolution – The resolution used when merging source analyzers

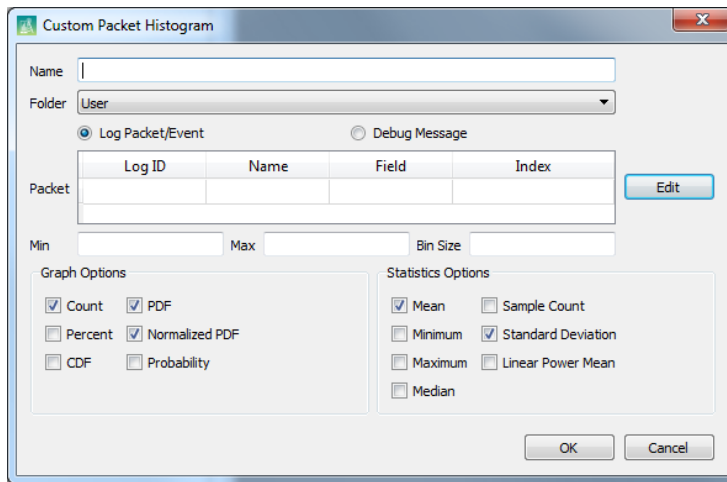
Other fields are automatically filled in from another dialog by clicking **New**.



- Analyzer – Selects the existing analyzer from the drop-down list
- Name/Column – The field name or column number in the selected analyzer

### 3.6.29.9 Creating a custom packet histogram

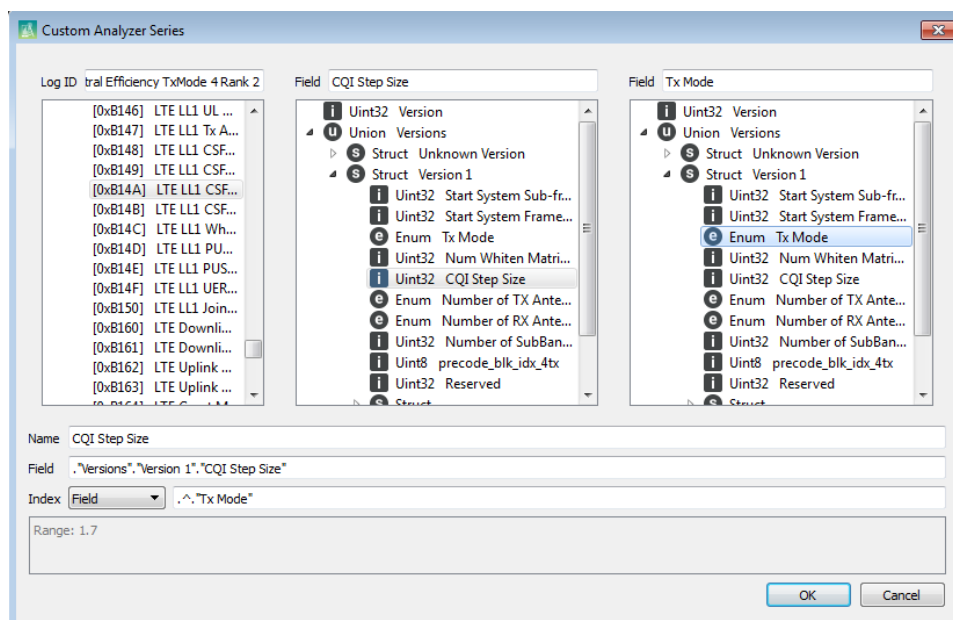
A custom histogram can be created based on a field in a log packet. To create a packet histogram, select **Analysis > New Packet Histogram** or right-click **User > New Packet Histogram**.



The fields in this dialog are:

- Name – The name of the custom histogram
- Folder – The output folder; defaults to user workspace
- Packet – These fields are filled in automatically when Edit is selected to bring up another dialog
- Min – The minimum value in the histogram; values smaller than min will be treated as min
- Max – The maximum value in the histogram; values larger than max will be treated as max
- Step – The step size in the histogram
- Options – Fields in this group specify the histogram property

Click **Edit** to bring up a dialog to specify a log packet field to create the histogram.

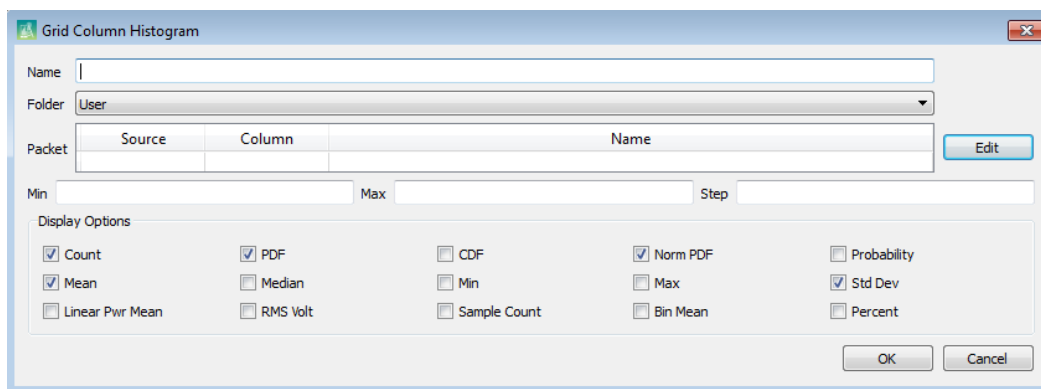


The fields that need to be specified are:

- Logs – Highlight the desired log packet
- Data field – Highlight the field from the tree view
- Index field – Index if the field is in an array

### 3.6.29.10 Creating a custom grid histogram

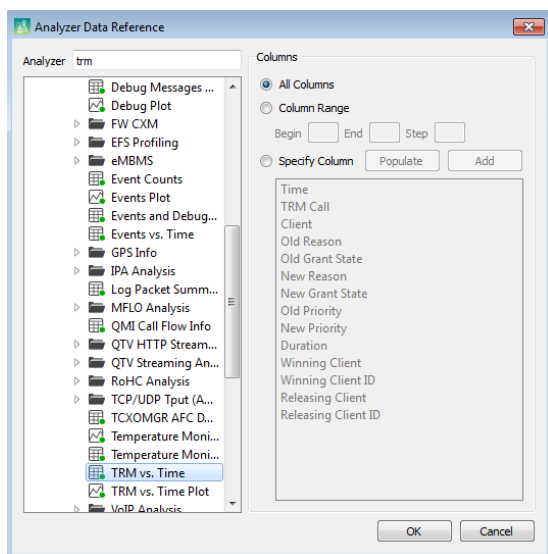
A custom histogram can be created based on a column in an existing grid analyzer. To create a grid histogram, select Analysis→New Grid Histogram or right-click User→New Grid Histogram.



The fields in this dialog are:

- Name – The name of the custom histogram
- Folder – The output folder; defaults to user workspace
- Packet – These fields are filled in automatically when Edit is selected to bring up another dialog
- Min – The minimum value in the histogram; values smaller than min will be treated as min
- Max – The maximum value in the histogram; values larger than max will be treated as max
- Step – The step size in the histogram
- Options – Fields in this group specify the histogram property

Click **Edit** to bring up a dialog to specify a log packet field to create the histogram.



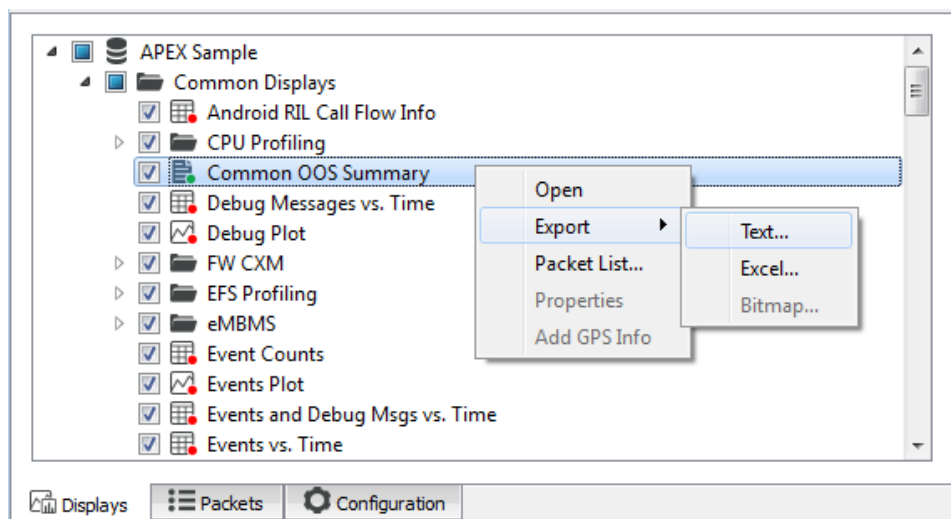
The grid analyzer can be specified from the drop-down list of an existing analyzer. Then specify either the column name or column number to create the histogram.

### 3.6.29.11 Exporting an analyzer

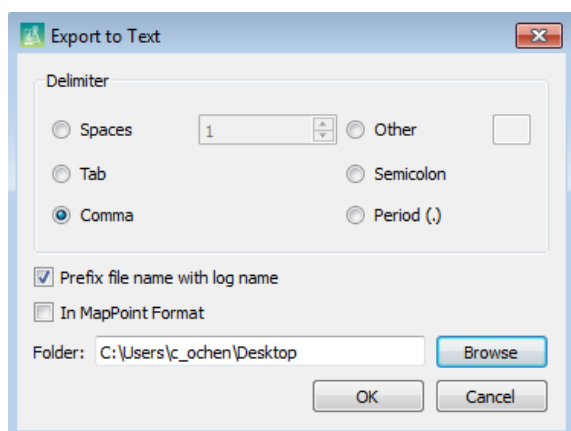
An analyzer listed in the left view can be exported to both a text file and an Excel file.

#### 3.6.29.11.1 Exporting to text

To export an analyzer or an entire folder of analyzers to a .txt file, either select Display from the menu bar or right-click the analyzer name, then select **Export to→Text**.



The following dialog appears, listing the text export options.



The delimiter for the data fields can be set. The default delimiter is Tab. A custom delimiter can also be set by selecting the other radio button and then entering the desired delimiter symbol.

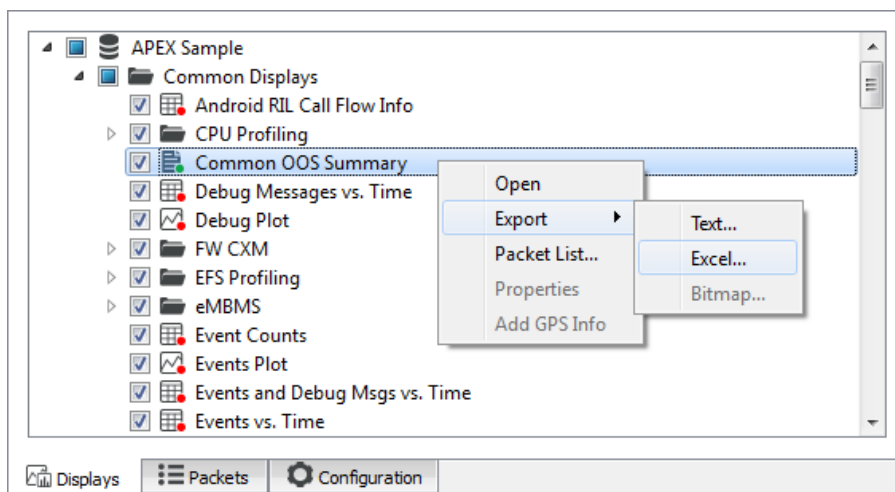
Select Generate Parsed Text File to create a text file containing the parsed output of all the packets in the log file. To have the Hex Dump and/or the MapPoint format included in the text file, select those options.

The path for the text file location can also be set. The default name of the text output file will be the name of the analyzer with a .txt extension. The text filename can be prefaced with the name of the log file from which the analyzer was generated by checking the appropriate box. This feature is useful when the same analyzer is generated and saved from multiple log files.

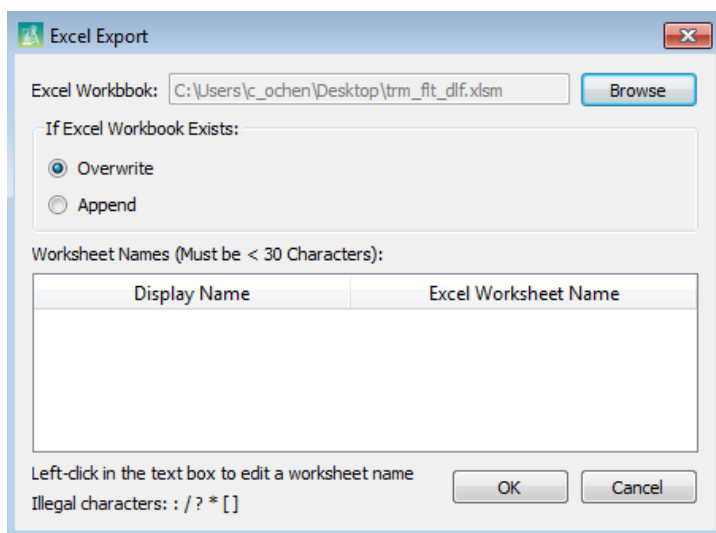


### 3.6.29.11.2 Exporting to Excel

To export an analyzer or an entire folder of analyzers to an Excel workbook file, select **Displays** from the menu bar or right-click the analyzer name. Then select **Export to→Excel**.



The following dialog appears, listing the Excel export options.



The path for the Excel workbook location can be set. The default name of the Excel workbook is the name of the log file from which the analyzer was generated, with an .xls extension. To change the workbook name, use **Browse** to select an existing workbook (.xls) name or type in a new name. If the workbook name already exists, either overwrite it or append it.

**NOTE:** An existing workbook must be closed before starting the export.

The default name of each worksheet in the workbook is the name of the analyzer from which it was generated. To change the worksheet name, left-click the name and edit it. For more information about the Excel workbook that is generated during the export function, see Section 6.2.4.

### 3.6.29.11.3 Exporting to Google Earth

QCAT can display the vs. Pos Map in Google Earth. The computer on which it is running requires Internet Explorer installed with the Google Earth plug-in enabled.

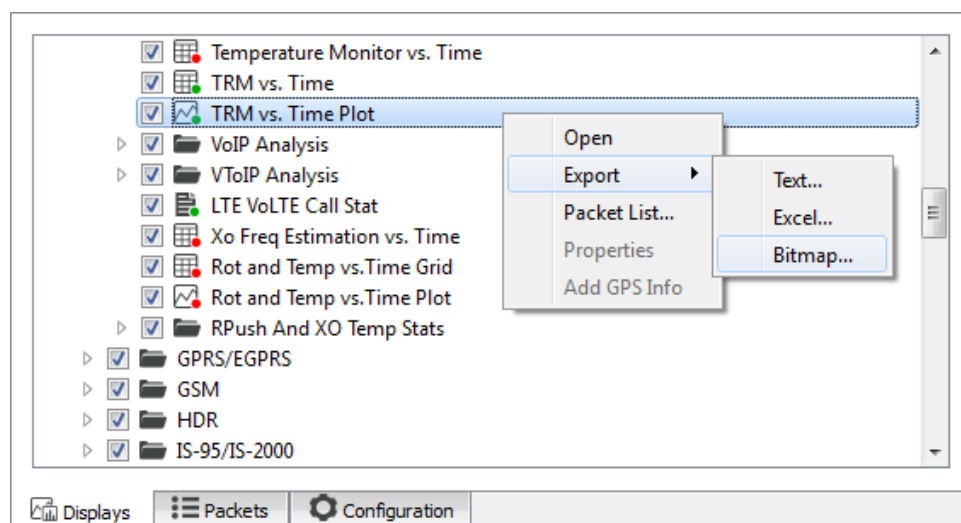
- Using the Google Earth plug-in, the generated map automatically moves the camera view to the position where the initial point of the data is located, and hence provides a sort of reference point for the path created by the data.
- The data points are color-coded based on the parameters set by the user or the analyzer and give the user an idea of how the data varies along the path of the plotted points.

Most analyzers by default do not contain map information. If the log file contains GPS information, a map can be generated by right-clicking any grid and selecting Add GPS Info. The resulting new grid will show up in the user workspace. After running the new grid, maps can easily be generated by right-clicking the column of a desired statistic and selecting Add Map.

**NOTE:** Google Earth becomes very slow with large sets of data. Sometimes using Add GPS Info and then setting the time resolution to be very low in its configuration will help enable Google Earth to handle the data.

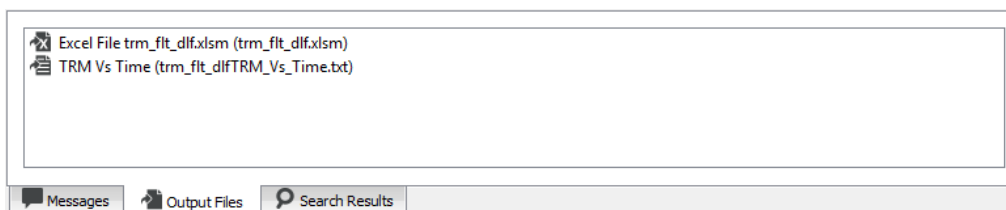
### 3.6.29.11.4 Exporting to a bitmap

To export an analyzer or an entire folder of analyzers to bitmap files, right-click the analyzer name. Then select Export to→Bitmap. This causes the analyzer to create the plot as it would be seen in the QCAT view, but saves it as a .png file.

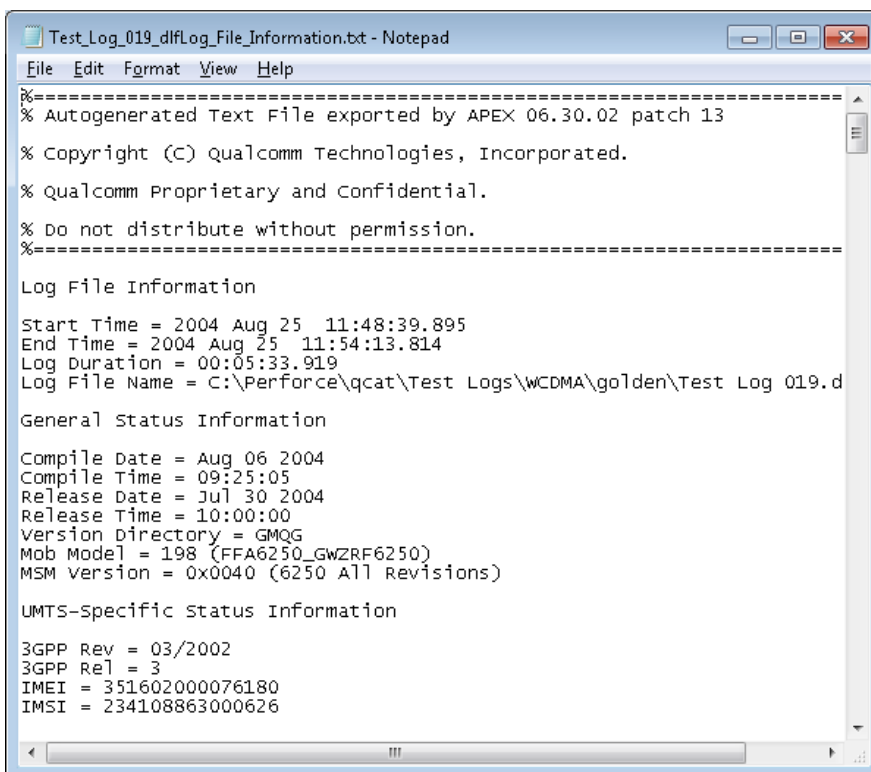


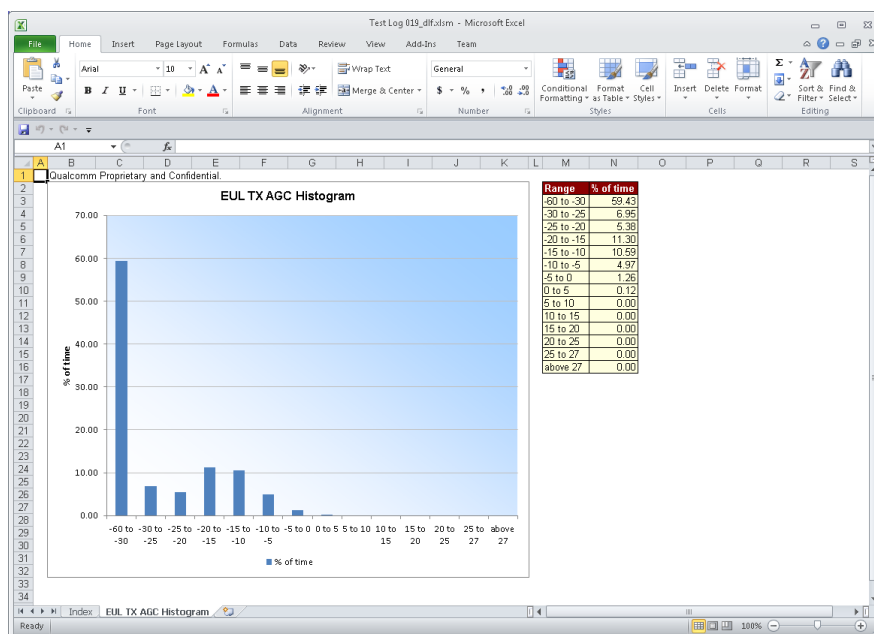
### 3.6.29.12 Viewing exported analyzer files

Exported files are listed in the bottom view in the Output Files tab as shown.

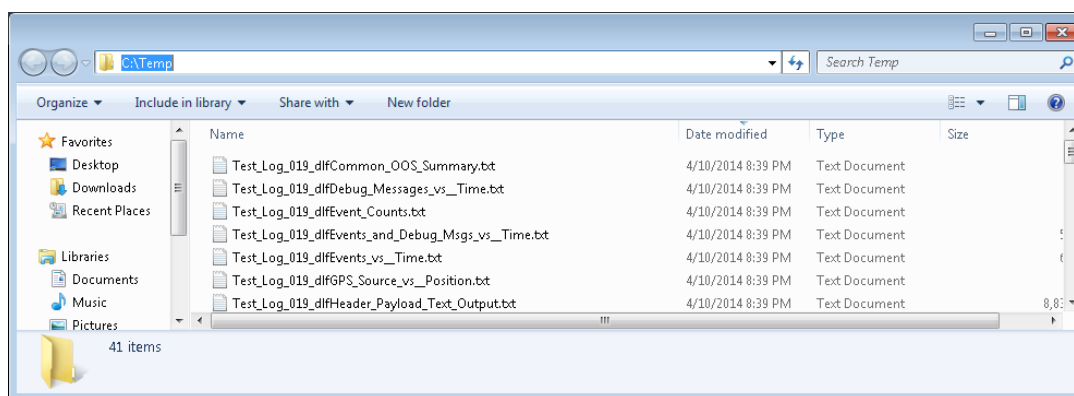


To view the contents of the text or Excel file, double-click the filename. This opens the file in either Notepad (.txt) or Excel (.xls).



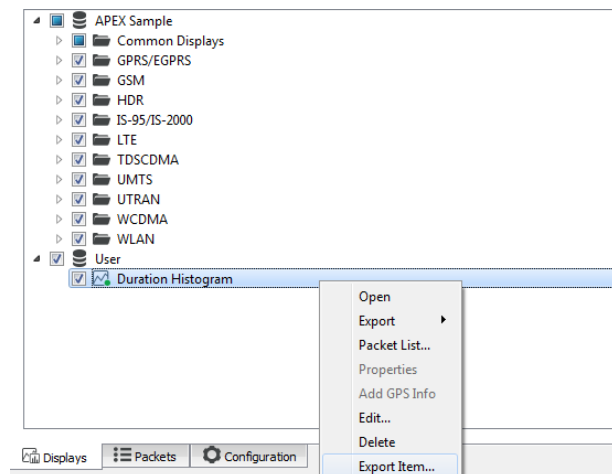


The Output Files tab also contains the names of the folders into which the .txt and .xls files were placed. To view the folder contents, double-click the folder name.



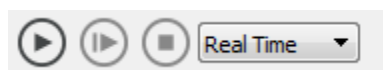
### 3.6.29.13 Exporting workspace items

Items in the user workspace can be exported so that analyzer definitions can be shared across multiple instances of QCAT. To save a workspace item, right-click it in the Displays tab and select Export Item.... This brings up the file save dialog. If multiple items are selected, they are all saved. This file can then be subsequently loaded into another instance of QCAT by either selecting File→Import Workspace Items... or by right-clicking a folder in the user workspace and selecting Import Item... The Export option in the Displays tab is shown here.



### 3.6.29.14 Analyzer Playback

Once an analyzer is open on the screen, it can be played back as though it were running in real time. This feature requires the analyzer to be a simple time display that does not rely on indexing or other logical requirements that prevent it from being displayed immediately, so some analyzers (like histograms, summaries, and some time plots) will not display output in the playback until the file finishes processing. Analyzers can be opened during playback and they begin showing data from the time of the playback when they were opened. Controls for playback are on the toolbar.



- Play/Pause – Plays or pauses the current playback
- Step – Steps the playback by one relevant packet
- Stop – Stops the current playback
- Replay speed – Gives an optional playback speed multiplier

## 4 Common Analyzers

This chapter describes informational analyzers that are common to all technologies. They provide information such as summary, general statistics, and debug messages, which are usually not plotted into graphs or maps. [Table 4-1](#) lists the common outputs generated by QCAT.

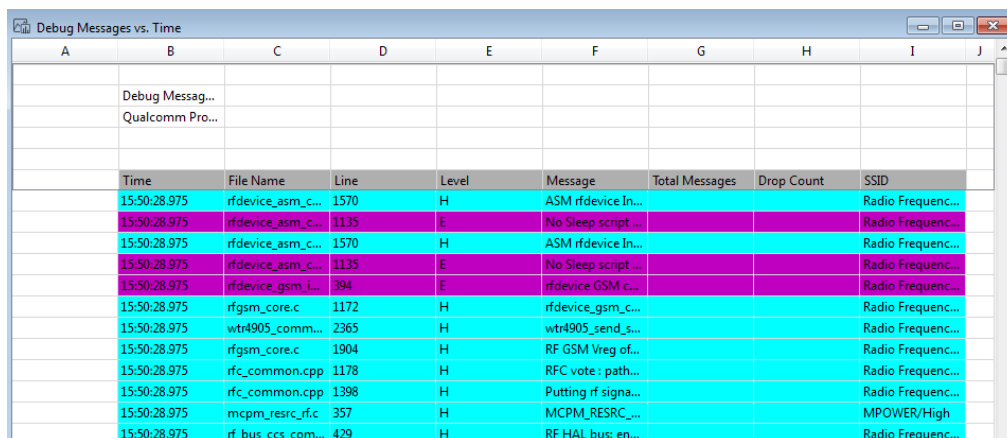
**Table 4-1 Common informational analyzers**

Workspace folder	Description	Display name/ Excel worksheet name
Common displays	Debug messages information	Debug messages vs. time
Common displays	Events and debug messages information	Events and debug messages vs. time
Common displays	Log file information	Log file information
Common displays	Log mask selection	Log mask selection
Common displays	Log packet summary	Log packet summary
Common displays	Events information	Events vs. time

### 4.1 Debug messages vs. time

This output file lists all the mobile-generated debug (F3) messages available in the processed log file. Listed with each message are its timestamp, source filename, source line number, and message level.

The log packets used for debug messages are extended debug message packet (log code 0x1FEB) and debug message packet (log code 0x1FEC).

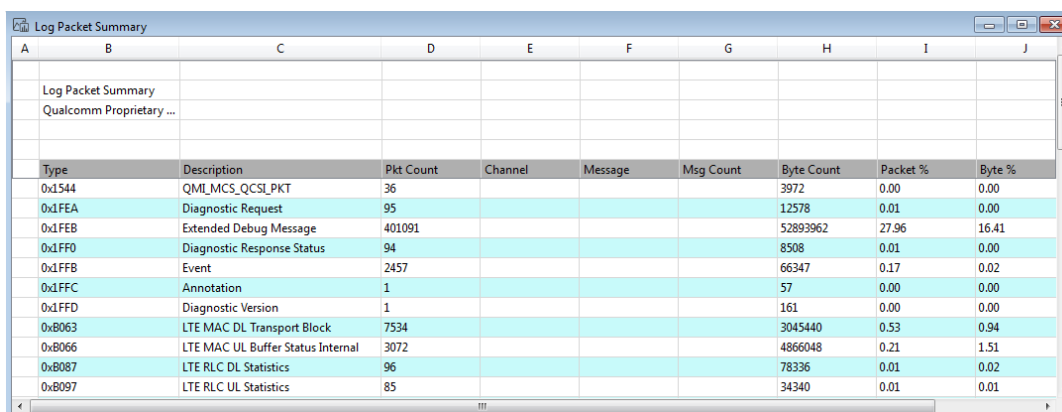


A	B	C	D	E	F	G	H	I	J
	Debug Messag...								
	Qualcomm Pro...								
	Time	File Name	Line	Level	Message	Total Messages	Drop Count	SSID	
	15:50:28.975	rfdevice_asm_c...	1570	H	ASM rfdevice In...			Radio Frequenc...	
	15:50:28.975	rfdevice_asm_c...	1135	E	No Sleep script ...			Radio Frequenc...	
	15:50:28.975	rfdevice_asm_c...	1570	H	ASM rfdevice In...			Radio Frequenc...	
	15:50:28.975	rfdevice_asm_c...	1135	E	No Sleep script ...			Radio Frequenc...	
	15:50:28.975	rfdevice_gsm_j...	394	E	rfdevice GSM c...			Radio Frequenc...	
	15:50:28.975	rfgsm_core.c	1172	H	rfdevice_gsm_c...			Radio Frequenc...	
	15:50:28.975	wtr4905_comm...	2365	H	wtr4905_send_s...			Radio Frequenc...	
	15:50:28.975	rfgsm_core.c	1904	H	RF GSM Vreg of...			Radio Frequenc...	
	15:50:28.975	rfc_common.cpp	1178	H	RF vote : path...			Radio Frequenc...	
	15:50:28.975	rfc_common.cpp	1398	H	Putting rf signa...			Radio Frequenc...	
	15:50:28.975	mcpm_resrc_rf.c	357	H	MCPM_RESRC...			MPOWER/High	
	15:50:28.975	rf_bus_ccs_com...	429	H	RF HAL bus: en...			Radio Frequenc...	



## 4.5 Log packet summary

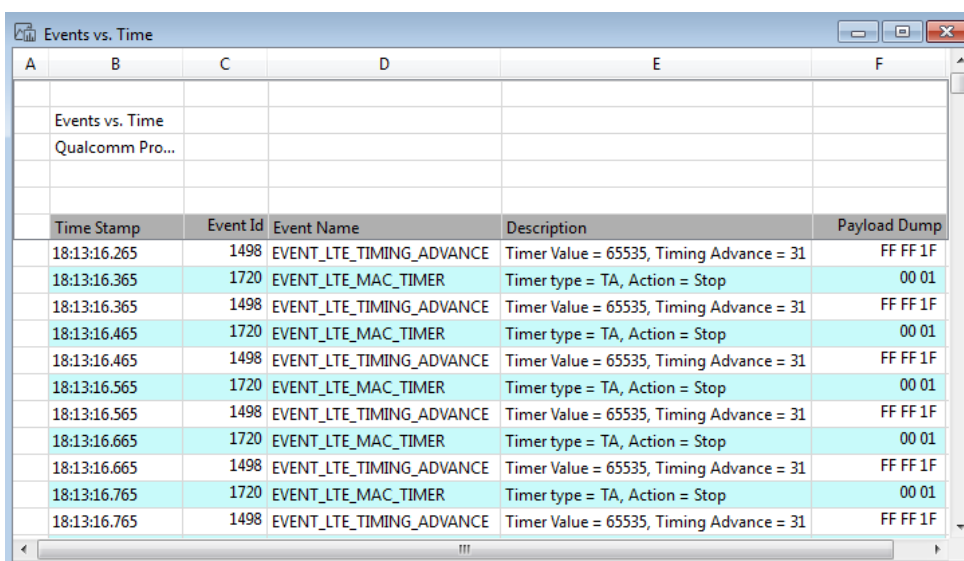
This output file provides a count of all log packet types in this log file.



Type	Description	Pkt Count	Channel	Message	Msg Count	Byte Count	Packet %	Byte %
0x1544	QMI_MCS_QCSL_PKT	36				3972	0.00	0.00
0x1FEA	Diagnostic Request	95				12578	0.01	0.00
0x1FEB	Extended Debug Message	401091				52893962	27.96	16.41
0x1FF0	Diagnostic Response Status	94				8508	0.01	0.00
0x1FFB	Event	2457				66347	0.17	0.02
0x1FFC	Annotation	1				57	0.00	0.00
0x1FFD	Diagnostic Version	1				161	0.00	0.00
0xB063	LTE MAC DL Transport Block	7534				3045440	0.53	0.94
0xB066	LTE MAC UL Buffer Status Internal	3072				4866048	0.21	1.51
0xB087	LTE RLC DL Statistics	96				78336	0.01	0.02
0xB097	LTE RLC UL Statistics	85				34340	0.01	0.01

## 4.6 Events vs. time

The Events vs. Time output gives a time-ordered list of decoded events as they are found in the log file. The log packet used is event 0x1FFB. The spreadsheet is shown here.



Time Stamp	Event Id	Event Name	Description	Payload Dump
18:13:16.265	1498	EVENT_LTE_TIMING_ADVANCE	Timer Value = 65535, Timing Advance = 31	FF FF 1F
18:13:16.365	1720	EVENT_LTE_MAC_TIMER	Timer type = TA, Action = Stop	00 01
18:13:16.365	1498	EVENT_LTE_TIMING_ADVANCE	Timer Value = 65535, Timing Advance = 31	FF FF 1F
18:13:16.465	1720	EVENT_LTE_MAC_TIMER	Timer type = TA, Action = Stop	00 01
18:13:16.465	1498	EVENT_LTE_TIMING_ADVANCE	Timer Value = 65535, Timing Advance = 31	FF FF 1F
18:13:16.565	1720	EVENT_LTE_MAC_TIMER	Timer type = TA, Action = Stop	00 01
18:13:16.565	1498	EVENT_LTE_TIMING_ADVANCE	Timer Value = 65535, Timing Advance = 31	FF FF 1F
18:13:16.665	1720	EVENT_LTE_MAC_TIMER	Timer type = TA, Action = Stop	00 01
18:13:16.665	1498	EVENT_LTE_TIMING_ADVANCE	Timer Value = 65535, Timing Advance = 31	FF FF 1F
18:13:16.765	1720	EVENT_LTE_MAC_TIMER	Timer type = TA, Action = Stop	00 01
18:13:16.765	1498	EVENT_LTE_TIMING_ADVANCE	Timer Value = 65535, Timing Advance = 31	FF FF 1F



# 5 Command Line Operations

---

QCAT 6.x provides command line access for a number of operations. The formats of the commands are described here.

## 5.1 Text parsing

The format of the text parsing command is as follows:

```
-txt [options] Logfile  
or  
-txt [options] Directory
```

If the name of a log file is given, then that file will be processed. If a directory is given, it will look for all .qmdl files within that directory, merge them, then parse the merged file to text.

### 5.1.1 Text parsing options

#### Filter

The optional filter option takes the path of a log packet filter that allows the user to filter out unwanted log packets. A log packet filter file can be created by running QCAT and saving a filter file as described in Section [3.6.9](#) of this document; e.g.:

```
-txt -filter=/user/filter.txt mylogfile.dlf
```

The filter file is a simple text file which alternatively, can be created and edited manually.

There are two forms of syntax:

### Inclusive syntax form

```
0
log code 1
log code 2
.....
-1
0
event code 1
event code 2
.....
-1
```

This form allows users to keep the intended log packets. The list items between 0 and -1 are intended to be kept. Each item starts on a new line. The first pair of 0 and -1 contains the list of log IDs to be kept, the second pair contains the list of event IDs to kept. The IDs are present in decimal format, eg.:

```
0
45339
8188
-1
0
456
1817
-1
```

In the above example, log ID 0xB11B (45339) and 0x1FFB (8188) are kept as well as event ID 456 and 1817.

## Exclusive syntax form

The exclusive syntax form serves the opposite function; it filters out unwanted packets:

```
1
log code 1
log code 2
.....
-1
1
event code 1
event code 2
.....
-1
```

This form allows users to filter out unwanted log packets. The list of items between 1 and -1 are intended to be filtered out. Each item starts on a new line. The first pair of 1 and -1 contains the list of log IDs to be filtered out; the second pair contains the list of event IDs to be filtered out. The IDs are present in decimal format, e.g.:

```
1
45339
8188
-1
1
456
1817
-1
```

In the above example, log ID 0xB11B (45339) and 0x1FFB (8188) are filtered out as well as event ID 456 and 1817.

## Property

The optional property (`-property=`) option allows you to change any configuration option in the application. The property name is enclosed in quotes, followed by a colon delimiter, followed by a value, e.g.:

```
-txt -property="Show Hex Dump":true mylogfile.dlf
```

## 5.2 Analysis output

The format of analysis output command is as follows:

```
-export [options] Logfile  
or  
-export [options] Directory
```

If the name of a log file is given, then that file will be processed. If a directory is given, it will look for all .qmdl files within that directory, merge them, then run analysis on the merged file.

### 5.2.1 Analysis output options

#### Filter

The optional filter option takes the path of a log packet filter that allows the user to filter out unwanted log packets. A log packet filter file can be created by running QCAT and saving a filter file as described in Section 3.6.9 of this document; e.g.:

```
-export -filter=/user/filter.txt mylogfile.dlf
```

#### Output

The optional output option allows the user to output the result to a specified location, e.g.:

```
-export -outputdir=/user/test mylogfile.dlf
```

#### Property

The optional property (-property=) option allows you to change any configuration option in the application. The property name is enclosed in quotes, followed by a colon delimiter, followed by a value, e.g.:

```
-export -property="Show Date in Analyzers":true mylogfile.dlf
```

#### Delimiter option

The optional delimiter (-delimiter=) option allows you to change what delimiter to use for the output text files. By default the delimiter is a tab, but -delimiter easily allows changing it to output csv files by specifying “,” as the delimiter, e.g.:

```
-export -delimiter="," mylogfile.dlf
```

## Workspace

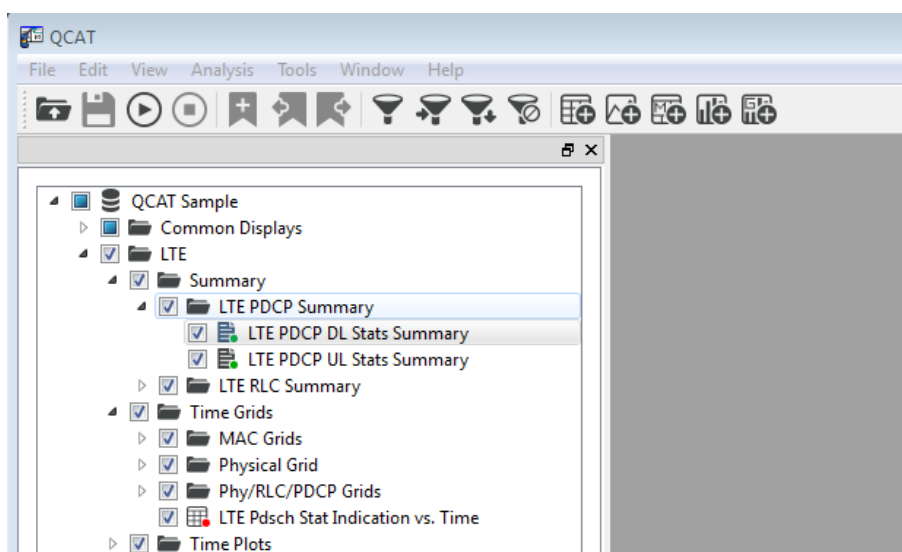
The workspace (`-workspace=`) option allows you to specify a workspace file to use instead of the default workspace loaded.

```
-export -analyzer="/user/MyWorkspace.aws" mylogfile.dlf
```

## Analyzer

The analyzer option lets the user specify which analyzer they want to run. The names of the analyzers are in the format Name of Workspace; subfolder; Name of Analyzer. For example, if a user wants to run LTE PDCP DL Stats Summary, as shown in [Figure 5-1](#), the command for this would be:

```
-export -analyzer="QCAT Sample;LTE;Summary;LTE PDCP Summary;LTE PDCP DL Stats Summary" mylogfile.dlf
```



**Figure 5-1 Sample path name of an analyzer**

Similarly, users can run all the analyzers under a folder, e.g.:

```
-export -analyzer="QCAT Sample;LTE;Summary" mylogfile.dlf
```

The above command runs all the analyzers under the Summary folder.

### 5.2.1.1 Workspace filter option

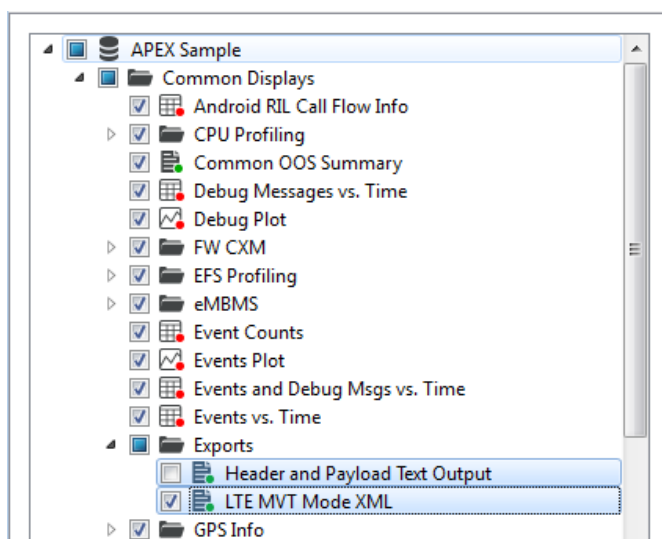
The workspace filter (`-wsfilter=`) option allows users to specify a set of analyzers or folders to be excluded or included in the run, e.g.:

```
-export -wsfilter=/user/wsfilterFile.txt mylogfile.dlf
```

A workspace filter file is a simple text file, which can be created and edited manually. The format of the file is:

```
QCAT Sample;Common Displays;Exports;Header and Payload Text Output $0
QCAT Sample;Common Displays;Exports;LTE MVT Mode XML $1
```

Figure 5-2 shows the two corresponding analyzers from the above example.



**Figure 5-2 Workspace Filter**

Here each line is the full path of the analyzer or folders, separated by semicolon, to be enabled or disabled, followed by the string “\$0” or “\$1”. In the above example, the Header and Payload Text Output analyzer is being disabled and the LTE MVT Move XML analyzer is being enabled in the run. The two most common use cases are enabling and disabling a set of selected analyzers.

#### Enabling a set of selected analyzers use case

This use case allows you to run a set of analyzers.

```
QCAT Sample $0
QCAT Sample;Summary;LTE Summary $1
QCAT Sample ;Summary;LTE TA Summary $1
```

In the above example, all analyzers in the QCAT Sample workspace are disabled, then only the LTE Summary and LTE TA Summary analyzers are enabled.

### Disabling a set of selected analyzers use case

This use case allows you to disable a set of analyzers.

```
QCAT Sample $1
QCAT Sample;Summary;LTE Summary $0
QCAT Sample ;Summary;LTE TA Summary $0
```

In the above example, all analyzers in the QCAT Sample workspace are enabled, then only the LTE Summary and LTE TA Summary analyzers are disabled.

## 5.3 Vocoder

The format of the vocoder command line support is as follows:

```
-vocoder[:codec option] [-outputdir=c:\xxx -reverse=0/1 -replace=0/1
-raw=0/1 -wav=0/1] Logfile
```

### 5.3.1 Vocoder options

```
-vocoder[:codec option]
```

The optional codec option ([:codec option]) is required only when there are vocoder packets (see Section 3.6.20 for log packet details) in the log file, and possible values are:

- 13k
- Auto
- amr-nb
- amr-wb
- eamr
- efr
- evrc
- evrc-b
- evrc-nw
- evrc-nw2k
- evrc-wb
- fr
- hr

- evs
- g711

```
[-outputdir=c:\xxx -reverse=0/1 -replace=0/1 -raw=0/1 -wav=0/1]
```

- -outputdir – Output directory; see [3.6.20.7](#) for the default path info;
- -reverse – Vocoder UI reverse byte order option; default is 0;
- -replace – Vocoder UI replace dropped frames option; default is 0
- -raw – Vocoder UI raw O/P option; default is 1
- -wav – Vocoder UI wav O/P option; default is 1

Example:

```
QCAT -vocoder c:\temp\pcmPacketOnlyLog.dlf
QCAT -vocoder:13k -outputdir=c:\Temp -replace=1 c:\Temp\testLog.dlf
QCAT -vocoder:evrc /Users/xxx/vocoderPacketLog.dlf
```

See Sections [3.6.20.6](#) and [3.6.20.7](#) for detailed information of generated files and output directory

## 5.4 ConvertQMDL

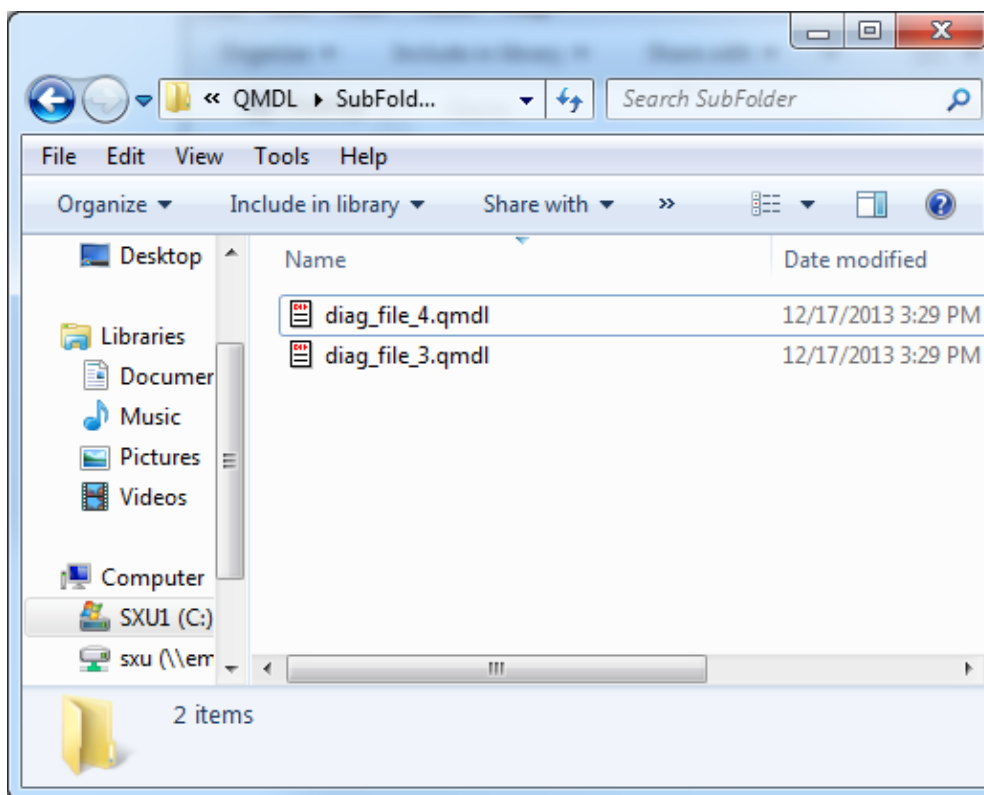
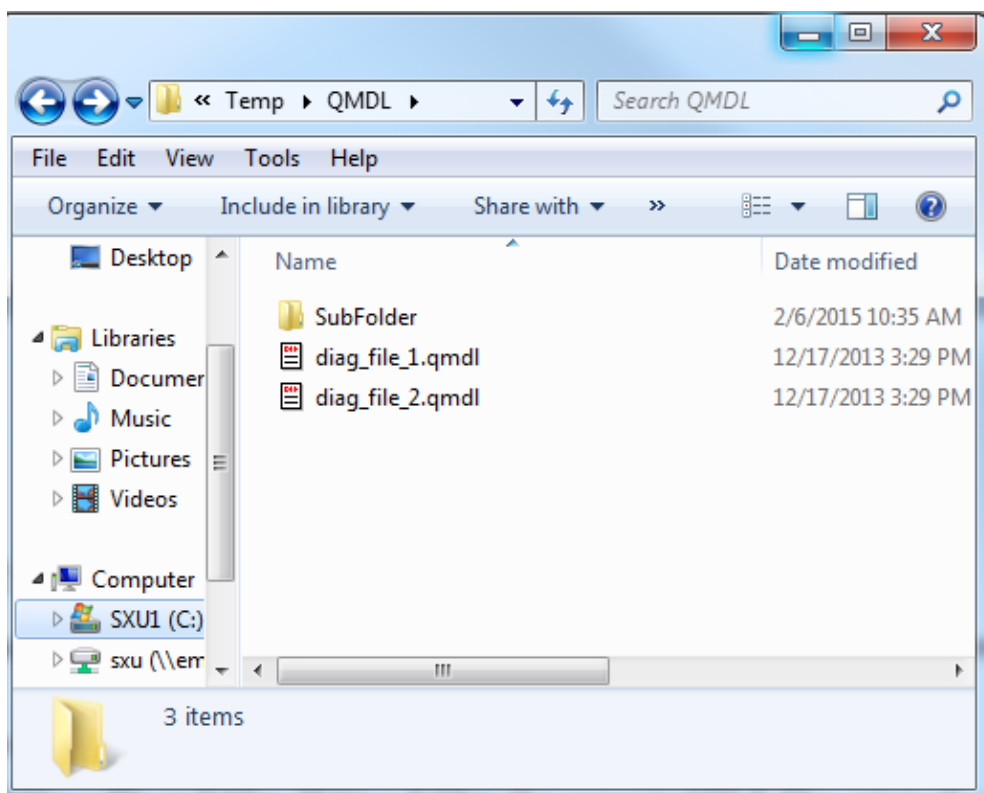
This command merges and puts the converted log files under the same folder where the .qmdl files reside. By default the files are converted to .isf; to convert to .dlf use the -dlf option.

```
QCAT -ConvertQMDL [options] f
or
QCAT -ConvertQMDL [options] f1 f2 ... fn
```

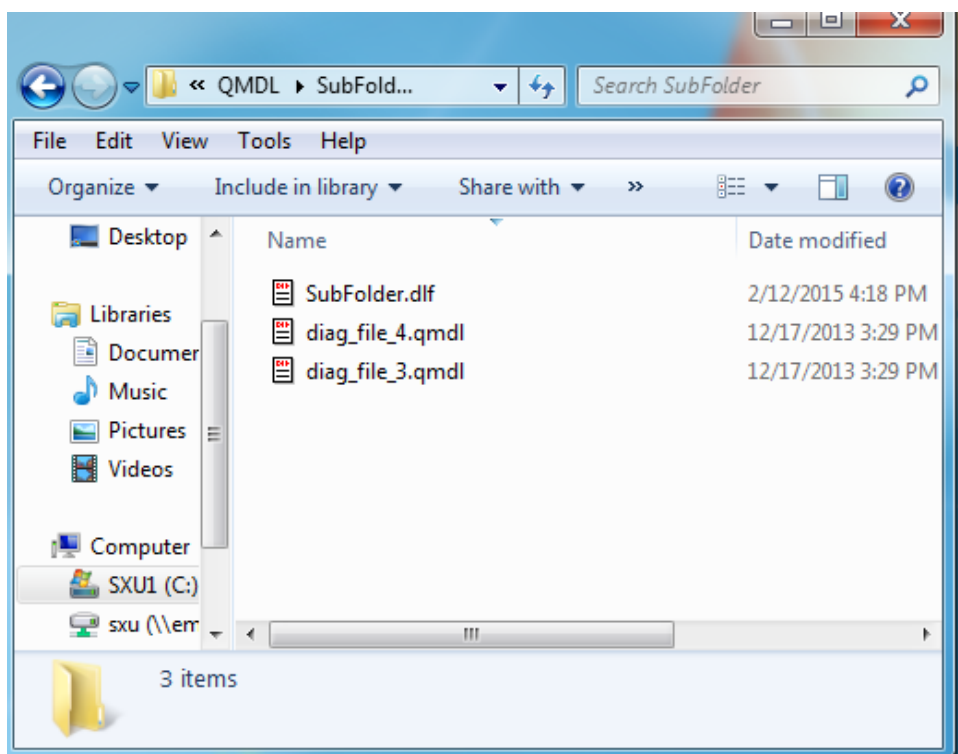
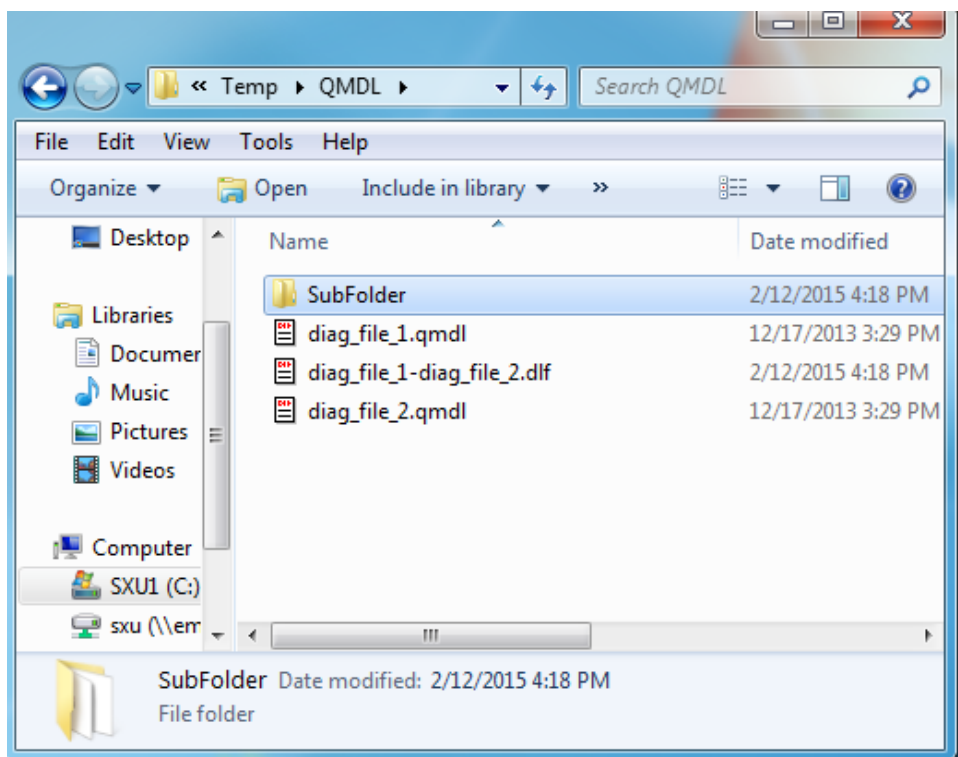
Where f, f1, f2 and fn could be a file or folder. If it's a folder, the converted .dlf files retain the original folder structure.



The original folder structure is shown below.



Converted .dlf files are shown below.



### 5.4.1 QShrink hash file directory option

QCAT will use the specified directory to load user QShrinkFiles instead of the default location (Documents/QCAT/QShrink).

```
-qshrinkdir=
```

By using the DLF output option, QCAT converts the .qmdl files to .dlf files rather than .isf files.

```
-dlf
```

## 5.5 Convert to .dlf

This command converts a file or directory of .qmdl files into a single .dlf. The .dlf file will be named the same as the original log file or the directory of the input but with a .dlf extension.

```
QCAT -dlf file  
or  
QCAT -dlf directory
```

### 5.5.1 .dlf conversion options

#### Filter

The optional filter option takes the path of a log packet filter that allows the user to filter out unwanted log packets. A log packet filter file can be created by running QCAT and saving a filter file as described in Section 3.6.9 of this document; e.g.:

```
-dlf -filter=/user/filter.txt mylogfile.isf
```

## 5.6 Convert to .isf

This command converts a file or directory of .qmdl files into a single .isf file. The .isf file will be named the same as the original log file or the directory of the input but with a .isf extension.

```
QCAT -isf file  
or  
QCAT -isf directory
```

### 5.6.1 .isf conversion options

#### Filter

The optional filter option takes the path of a log packet filter that allows the user to filter out unwanted log packets. A log packet filter file can be created by running QCAT and saving a filter file as described in [Section 3.6.9](#) of this document; e.g.:

```
-isf -filter=/user/filter.txt mylogfile.dlf
```

## 6 Scripting with QCAT (Windows Only)

QCAT 6.x provides a COM-based automation interface that allows scripts written in languages such as Perl, VBScript, and JavaScript to access features of the application.

### 6.1 Sample scripts

The QCAT installation installs sample Perl scripts, listed in [Table 6-1](#), in a folder called Script in the installation path.

**Table 6-1 Listing of sample scripts**

File	Description
GetVersion.pl	Invokes QCAT, get Version, and sets QCAT to be visible
ConfigTest.pl	Prints a list of all available configuration properties Tests getting and setting configuration properties by name
OptionsTest.pl	Manipulates application options
SimpleParse.pl	Single-step log file processing and automation error handling
FilterSample.pl	Sets the packet filter to load only debug messages and processes a log file, then saves as text
SortedIterationSample.pl FilteredIterationSample.pl	Gets a packet iterator and then iterates through a few packets and prints their contents
ProcessDirectory.pl	Creates the parsed text file for each file in a directory
ProcessPacket.pl	Sends a buffer of raw bytes to QCAT to be interpreted as a log packet, which can then be parsed
ProcessQmdlPacket.pl	Sends a buffer of raw bytes to QCAT to be interpreted as a qmdl payload, which can then be parsed
GetPacketList.pl	Gets the list of packets supported by QCAT
GetVocoderPCM.pl	Generates the vocoder PCM output files
SplitLog.pl	Splits a specified log file into a set of smaller log files by specifying either how many smaller files to make or a time duration for each file
closeFile.pl	Closes the log file
TimeWindowSample.pl	Sets the time window to a fraction of its original length and saves as text
UpdateQCAT4License.pl	Updates the QCAT 4.x license from the QCAT 6.x license
DebugMsgFilter.pl	Sets the debug message filter and processes a log file, then saves as text
EventsFilter.pl	Sets the event filter and processes a log file, then saves as text
MergeFusionLogs.pl	Merges two QCAT parsed outputs based on timestamp to show the synchronized information from two targets in Fusion mode
MergeWithWireshark.pl	Merges QCAT parsed text with wireshark text output
GetLicesnseTimeRemaining.pl	Displays the amount of time before the installed version's license expires

File	Description
SaveAsDlf.pl	Shows an example of how to save the current file as a new .dlf file
GetField.pl	Shows how to use the FieldExists, GetField, and GetFieldArray interfaces
ExportTextAndExcel.pl	Shows how to export analyzers to text and Excel outputs
ExtractPcmFiles.pl	Used internally by QCAT
ExtractVocoderData.pl	Used internally by QCAT
qcatobj.pl	Used internally by QCAT
tsharkobj.pl	Used internally by QCAT
tsharkverboseobj.pl	Used internally by QCAT

**NOTE:** One or more of the sample scripts assumes the presence of a log file in C:\Temp\Sample.dlf. The QCAT installation and the scripts do not create or copy any such log file in the Temp folder. The scripts require you to copy any log file under the Sample.dlf name to the specified path.

## 6.2 QCAT automation objects

QCAT distributes responsibilities between several automation objects. The top-level object is the QCAT6.Application object.

### 6.2.1 QCAT6.Application object

This is the main application object. Creating this object invokes the QCAT application. The application terminates when the script releases this object.

The following code demonstrates how to create the application object:

```
use Win32::OLE;
my $qcat_app = new Win32::OLE 'QCAT6.Application';
if(!$qcat_app)
{
    print "ERROR: Unable to invoke the QCAT application.\n";
    die;
}
```

The following code demonstrates how to release the application object:

```
$qcat_app = NULL;
```

### 6.2.1.1 Properties

#### **BOOL AnnotateSILK [Put/Get]**

- True – OTA messages processed by SILK will include annotations
- False – SILK output will not be annotated

#### **String AppVersion [Get]**

This is the application version string in the format 0n.nn.nn.

#### **LONG BandClass [Put/Get]**

This explicitly sets the band class to use single-packet parsing (see ProcessPacket method).

#### **BOOL DatabaseStatus [Get]**

- True – Event database is available and will provide event details (requires QXDM)
- False – Event database was not found

#### **IAutoLogPacket\* FirstPacket [Get]**

This is the packet automation object that wraps the first packet in the log file.

#### **String LastError [Get]**

This is the description of the last error encountered.

#### **LONG LastErrorCode [Get]**

This is unused.

#### **LONG LogStartTime [Get]**

This is the start time of the log as the number of seconds since midnight Jan 1, 1970 (UTC).

#### **LONG LogEndTime**

This is the end time of the log as the number of seconds since midnight Jan 1, 1970 (UTC).

#### **Double LogDuration [Get]**

This is the time span of the log file in seconds.

#### **LONG Mode [Put/Get]**

This explicitly sets the Timestamp mode for single-packet parsing (see ProcessPacket method):

- UNKNOWN\_MODE – 0
- CDMA\_MODE – 1
- WCDMA\_MODE – 4
- GSM\_MODE – 5

**LONG Model [Put/Get]**

This explicitly sets the model for single-packet parsing (see ProcessPacket method).

**LONG PacketCount [Get]**

This is the number of packets in the currently open log file.

**LONG VisiblePacketCount [Get]**

This is the number of visible packets in the currently open log file. Visible packets are ones that are neither filtered out nor outside of the active time window.

**AutoPacketFilter PacketFilter [Get]**

This is the filter automation object for removing packets by type code.

**LONG PRev [Put/Get]**

This explicitly sets the PRev to be used by SILK in decoding OTA messages.

**LONG PilotInc [Put/Get]**

This explicitly sets the Pilot Inc to be used by SILK in decoding OTA messages.

**BOOL ShowHexDump [Put/Get]**

- True – Print packet hex dump in addition to the pretty-print
- False – Do not print the hex dump

**String SILKVersion [Get]**

This is the SILK revision string in the format n.nn.

**BOOL SplitEvents [Put/Get]**

- True – Split multievent packets into single-event packets
- False – Leave multievent packets bundled

**VARIANT SupportedPackets [Get]**

This returns an array of short (16-bit) type codes; these type codes are the packet types supported by QCAT.

**LONG StateRequired([in] LONG packetType) [Get]**

This returns a bitmask that identifies the log state information that must be set in order to correctly process a packet of the given type (see ProcessPacket method):

- REQUIRES\_NONE – 0x00000000
- REQUIRES\_MODE – 0x00000001 (see Mode property)
- REQUIRES\_P\_REV – 0x00000002 (see PRev property)
- REQUIRES\_BANDCLASS – 0x00000004 (see BandClass property)
- REQUIRES\_MODEL – 0x00000008 (see Model property)



**String TimestampFormat [Put/Get]**

This sets/gets the timestamp display format; valid values are:

- Default – Default QCAT 4.x/5.x/6.x time formatting
- Calendar – Displays all dates as calendar dates
- Days – Displays a number of days rather than a date

**String TimestampLocale [Put/Get]**

This sets/gets the timestamp locale; valid values are:

- Default – Default QCAT 4.x/5.x/6.x time locale
- UTC – All times will be printed as UTC time
- Local – All times will be adjusted to the local time zone

**BOOL UseT53Standard [Put/Get]**

This controls use of the T53 standard for decoding OTA messages.

**BOOL UsePCTime [Put/Get]**

- True – Attempts to align the timestamps of the log packets with the PC system time
- False – Uses log packet timestamps as they are

**BOOL V3xCompatibility [Put/Get]**

- True – Uses QCAT 3.x version packet names
- False – Uses QCAT 4.x+ naming (recommended)

**BOOL Visible [Put/Get]**

This controls whether the UI is visible.

**6.2.1.2 Methods****void Force3GPPRev**

```
(  
[in] short nMonth,  
[in] short nYear  
);
```

This explicitly sets the 3GPP Revision for parsing WCDMA RRC Signaling messages. This is only necessary when the correct 3GPP revision cannot be determined from the log file.

## BOOL GenerateVocoderPCM

```
(  
[in] BSTR txOutPath,  
[in] BSTR rxOutPath,  
[in] LONG mode  
);
```

This extracts vocoder frames from log packets, concatenates, and converts them to a PCM file that can be played back using a PCM audio utility. Descriptions are:

- txOutPath – Output file path for the Tx stream
- rxOutPath – Output file path for the Rx stream
- Mode – Vocoder mode if not contained in the log packets; must be one of the following:
  - Auto Select (0) – Determine from log file if possible
  - AMR-NB(1)
  - EFR(2)
  - FR(3)
  - HR(4)
  - EVRC(5)
  - 13K(6)
  - AMR-WB(7)
  - EVRC-B(8)
  - EVRC-WB(9)

## BOOL ExtractVocoderFrameFiles

```
(  
[in] BSTR txOutPath,  
[in] BSTR rxOutPath,  
[in] LONG mode  
  
);
```

- txOutPath – Output path (a directory) for Tx frames
- rxOutPath – Output path (a directory) for Rx frames
- Mode – Vocoder mode if not contained in the log packets (see GenerateVocoderPCM for values)

This extracts the vocoder frames into files and converts them using the correct C-SIM. Frame files and C-SIM output files are saved to the given directory paths. See Section [3.6.20.8](#).

## BOOL ExtractPcmAudio

```
(  
[in] BSTR txOutPath,  
[in] BSTR rxOutPath  
);
```

- txOutPath – Output file path for the Tx stream
- rxOutPath – Output file path for the Rx stream

This extracts PCM samples from the vocoder frames into files and converts them using the correct C-SIM. Frame files and C-SIM output files are saved to the given directory paths. See [Section 3.6.20.4](#).

## VARIANT GetProperty

```
(  
[in] BSTR name  
);
```

This gets a configuration property by name. It returns empty if the property was not found.

## String GetPacketTypeName

```
(  
[in] LONG type  
);
```

- Type – Packet type code to look up. This returns the English name for the type.

## String GetPropertyList

```
(  
[in] BOOL bIncludeValues  
);
```

This returns a list of all the configuration properties in the application. These properties can be set/get using PutProperty/GetProperty.

- bIncludeValues – Includes the current value for the property in the returned string

## BOOL OpenLog

```
(  
[in] BSTR pathName  
);
```

This opens a log file. It returns TRUE on success. If the result is FALSE, it retrieves the last error message from the LastError property.

- pathName – Full path to the log file to be opened

## BOOL closeFile( )

This closes the log file.

## BOOL Process

```
(  
[in] BSTR pszLogPath,  
[in] BSTR pszOutPath,  
[in] BOOL bHexDump,  
[in] BOOL b3xCompatibility  
);
```

This opens a log file and pretty-prints the packets into an ASCII text file. Descriptions are:

- BSTR pszLogPath – Full path to the target log file
- BSTR pszOutPath – Full path of the text file to produce
- BOOL bHexDump – If TRUE, the hex dump for each packet will be printed in the output file
- BOOL b3xCompatibility – If TRUE, the QCAT 3.x version of the packet name will be used (FALSE is recommended for new scripts)

## BOOL ProcessDebug

```
(  
[in] BSTR pszLogPath,  
[in] BSTR pszOutPath  
);
```

This opens a log file and pretty-prints only the debug message packets into an ASCII text file. Descriptions are:

- BSTR pszLogPath – Full path to the target log file
- BSTR pszOutPath – Full path of the text file to produce

## String ParsePackets

```
(
[in] LONG start,
[in] LONG count
);
```

Parses a range of packets within the current file into text.

Descriptions are:

- LONG start – First packet to parse to text
- LONG count – Total number of packets to parse to text (starting from “start”)

## AutoLogPacket ProcessPacket

```
(
[in] VARIANT packet,
);
```

This wraps a raw DLF packet payload with an AutoLogPacket automation object (see AutoLogPacket).

- Packet – Full DLF packet as a byte array (VT\_I1 | VT\_ARRAY)

This returns an AutoLogPacket automation object.

**WARNING:** *Some packets require state information from other log items and will not process correctly without it.* If the packet type requires missing state information for proper processing, the return value is NULL and the LastError property will list the missing state information. To set missing state parameters, use the BandClass, Mode, Model, and PRev properties, and the Force3GPPRev method.

## AutoLogPacket ProcessQmdlPacket

```
(
[in] VARIANT packet,
);
```

This wraps a raw QMDL packet payload with an AutoLogPacket automation object (see AutoLogPacket).

- Packet – Full QMDL packet as a byte array (VT\_I1 | VT\_ARRAY) including the HDLC CRC and termination byte (0x7E)

This returns an AutoLogPacket automation object.

**WARNING:** *Some packets require state information from other log items and will not process correctly without it.* If the packet type requires missing state information for proper processing, the return value is NULL and the LastError property will list the missing state information. To set missing state parameters, use the BandClass, Mode, Model, and PRev properties, and the Force3GPPRev method.

## BOOL PutProperty

```
(  
[in] BSTR name,  
[in] VARIANT value  
);
```

This sets a configuration property by name. Valid configuration property names can be listed using the GetPropertyList method. Descriptions are:

- Name – Name of the property to be set
- Value – New value for the property

This returns FALSE if the property with the given name is not found.

## BOOL SaveAsText

```
(  
[in] BSTR outPath  
);
```

This parses the current log file and saves the output text to the file specified by pszOutPath.

- outPath – Output file path

## BOOL SaveAsDLF

```
(  
[in] BSTR outPath  
);
```

This saves the current log file as a new DLF.

- outPath – Output file path

## BOOL SaveAsUpdatedDLF

```
(
[in] BSTR outPath
);
```

This saves the current log file as a new DLF, updating the timestamps to be as they are displayed. This means saving the PC Time offset into the log headers.

- outPath – Output file path

## BOOL SaveAsISF

This saves the current log file as a new .isf file.

```
(
[in] BSTR outPath
);
```

Parameter description is:

- outPath – Output file path

## BOOL SaveAsUpdatedISF

This saves the current log file as a new .isf file, updating the timestamps to be as they are displayed. This means saving the PC time offset into the log headers.

```
(
[in] BSTR outPath
);
```

Parameter description is:

- outPath – Output file path

## void SetTimeWindow

```
(
[in] double offset,
[in] double length
);
```

This limits packets to a specified time window. Descriptions are:

- Offset – Beginning of the window as an offset from the first packet in the log (in seconds)
- Length – Size of the window in seconds

## **void SetTimeWindowAbsolute**

This limits packets to a specified time window.

```
(  
[in] BSTR startTime,  
[in] BSTR endTime  
);
```

Parameter descriptions are:

- **startTime** – A string containing the time at which to start the time window
- **endTime** – A string containing the time at which to end the time window

The string for the **startTime** and **endTime** is to be formatted as “yyyy/mm/dd hh:mm:ss.ddd”, e.g., “2013/01/05 12:05:23.123”.

## **void SetPacketWindow**

```
(  
[in] long offset,  
[in] long length  
);
```

This limits packets to a specified window. Descriptions are:

- **Offset** – The number of packets to be clipped off the front of the log
- **Length** – The number of packets to be viewed

## **BOOL SortByTime();**

This sorts the packets of the open log file by timestamp.

## **BOOL SortByIndex();**

This sorts the packets of the open log file by file index.

## **BOOL SortByLogId();**

This sorts the packets of the open log file by log ID.

## **BOOL SortByLogName();**

This sorts the packets of the open log file by log name.

## **BOOL SortBySize();**

This sorts the packets of the open log file by log size.



## 6.2.2 QCAT6.AutoLogPacket

### 6.2.2.1 Properties

#### **Unsigned Short Type [Get]**

This is the packet type code.

#### **Short Length [Get]**

This is the length of the packet in bytes (including DLF item header – 12 bytes).

#### **String Name [Get]**

This is the name for this type of packet.

#### **String TimestampAsString [Get]**

This is the timestamp as a date/time string.

#### **String Text [Get]**

This is the parsed payload of the packet as would appear in the QCAT text output.

#### **Unsigned long SortedIndex [Get]**

This is the time sorted index of the packet and can be used in conjunction with functions taking a packet index, i.e., SetPacketWindow.

#### **String Subtitle [Get]**

This is the subtitle of this packet. For events this is the event name. For debug messages this is the message name. For OTA packets it contains information about the payload.

### 6.2.2.2 Methods

#### **Bool Next();**

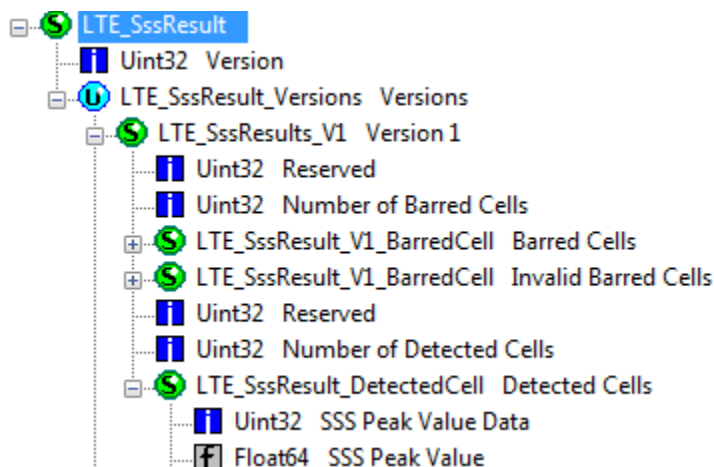
This goes to the next packet. It returns FALSE if this is the last packet.

#### **Bool FieldExists();**

This returns TRUE when a string specifier references a valid field within the packet. The string specifier is described in GetField().

## String GetField();

This returns the first instance of data where the field name matches the string specifier within the packet. The string specifier is a portion of the log structure definition, where as much or as little of the structure can be specified as desired, with as little required as the field name. It can also specify indexes in arrays, i.e., given the following structure:



Example accessors are:

- “.SSS Peak Value” – Will match every instance of SSS Peak Value in the packet
- “.SSS Peak Value”[0] – Will match only instances of SSS Peak Value at array index 0
- “.Detected Cells”.SSS Peak Value” – Will only match SSS Peak Value that exists within a field called Detected Cells
- “.Detected Cells”[1]. “SSS Peak Value” – Will only match SSS Peak Value that exists within a field called Detected Cells where Detected Cells is at index 1

## SafeArray(String) GetFieldArray();

This returns every instance of data where the field name matches the string specifier within the packet. The string specifier is described in GetField().

## 6.2.3 QCAT6.AutoPacketFilter

### 6.2.3.1 Methods

#### Void SetAll

```
(
  BOOL bEnable
);
```

This enables/disables all packet types.

- bEnable
  - True – Enable
  - False – Disable

#### Void SetRange

```
(
  short nFirst,
  short nLast,
  BOOL bEnable
);
```

This enables/disables a range of packet types (inclusive). Descriptions are:

- nFirst – First packet type to be enabled/disabled
- nLast – Last packet type to be enabled/disabled
  - True – Enable
  - False – Disable

#### Void Set

```
(
  short type,
  BOOL bEnable
);
```

This enables/disables a single packet type. Descriptions are:

- type – Type of packet to disable
- bEnable
  - True – Enable
  - False – Disable

## BOOL IsEnabled

```
(  
short type  
);
```

This returns TRUE if the given packet type is enabled.

- Type – Packet type to query

## Void Commit();

This applies the current filter to the log file.

## 6.2.4 QCAT6.AutoWorkspace

### Methods

#### BOOL ExportToExcel

This runs all analyses selected in the workspace and stores the results in an Excel workbook.

```
(  
[in] BSTR strOutputPath,  
[in] BOOL bAppend  
);
```

Parameter descriptions are:

- strOutputPath – Excel workbook output path
- bAppend – Appends the data to an existing workbook

OutputPath is the absolute path name.

#### BOOL ExportToText

This runs all analyses selected in the workspace and stores the results in text files with data columns delimited by CellDelimiter (usually the tab character “\t”).

```
(  
[in] BSTR strOutputPath,  
[in] BSTR strCellDelimiter  
);
```

OutputPath is the absolute path name.

## BOOL ExportToTextAndExcel

This runs all analyses selected in the workspace and stores the results in text files with data columns delimited by CellDelimiter (usually the tab character “\t”) as well as in an Excel file.

```
(  
    String strOutputPath,  
    String strTextCellDelimiter,  
    Bool bAppendExcel  
);
```

OutputPath is the absolute path name.

## BOOL ExportToBitmap

This runs all analyzers selected in the workspace and stores the resulting graphs in bitmap files.

```
(  
[in] BSTR strOutputPath  
);
```

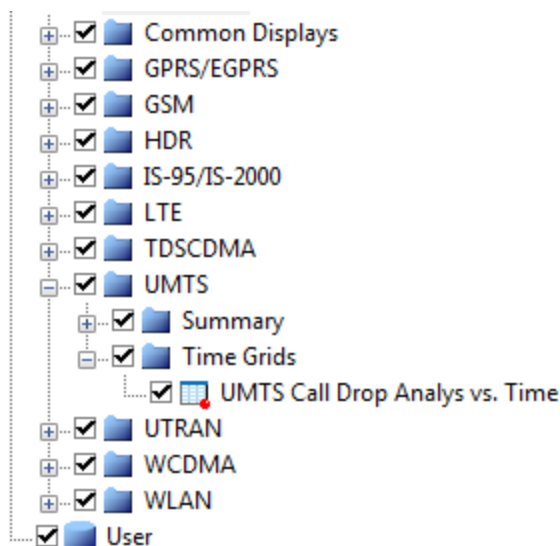
OutputPath is the absolute path name.

## BOOL SelectOutput

This selects a display or a folder of displays (by path) in the workspace to be exported.

```
(  
[in] BSTR strPath,  
[in] BOOL bEnable  
);
```

To select a workspace element, concatenate the names of all parent folders with the item's name separated by semicolons. A lone semicolon represents the root.



For example, the path to UMTS Call Drop would be:

```
;UMTS;Time Grid;UMTS Call Drop Analysis vs. Time
```

### **SAFEARRAY(unsigned short) GetRequiredPackets**

This returns an array of the log codes used by the specified analyzer.

```
(
[in] BSTR strPath
);
```

The specifier is the same as that used in SelectOutput.

### **SAFEARRAY(unsigned short) GetRequiredEvents**

This returns an array of the event IDs used by the specified analyzer.

```
(
[in] BSTR strPath
);
```

The specifier is the same as that used in SelectOutput.

# 7 Scripting with QCAT (Mac and Linux Only)

---

QCAT 6.x provides a D-Bus-based automation interface that allows scripts written in languages such as Perl to access features of the application.

## 7.1 Sample scripts

Table 7-1 lists the sample Perl scripts included in the QCAT installation; these scripts are located in the Script folder under the installation path.

**Table 7-1 List of sample scripts**

File	Description
GetVersion.pl	Invokes QCAT, gets version, and sets QCAT to be visible
OptionsTest.pl	Manipulates application options
SimpleParse.pl	Single step log file processing and automation error handling
Set3GPPRev.pl	Checks if the events database was loaded and forces a 3GPP rev value (only valid for WCDMA/GSM/UMTS logs)
ExportText.pl	Loads a workspace and exports analyzer files to text
FilterSample.pl	Sets the packet filter to load only debug messages, processes a log file, and saves the log file as text
SortedIterationSample.pl FilteredIterationSample.pl	Gets a packet iterator, iterates through a few packets, and prints their contents
ProcessDirectory.pl	For each file in a directory, prints the parsed text file and exports the analyzers to text
ProcessPacket.pl	Sends a buffer of raw bytes to QCAT to be interpreted as a log packet, which can then be printed
ConfigTest.pl	Gets and sets CDMA everything vs time resolution
GetPacketList.pl	Prints the list of packet types supported by QCAT
GetVocoderPCM.pl	Generates the vocoder PCM output files
ProcessTextAndDebugMsgFiles.pl	Processes all the .dlf files in INPUT_DIRECTORY and exports the text for all ENABLED_OUTPUTS (optional Excel output)
TimeWindowSample.pl	Sets the time window to a fraction of its original length and saves the time window as text
SplitLog.pl	Splits a specified log file into a set of smaller log files by specifying either how many smaller files to make or a time duration for each file
closeFile.pl	Closes the log file
DebugMsgFilter.pl	Sets the debug message filter, processes a log file, and saves the log file as as text

File	Description
EventsFilter.pl	Sets the event filter, processes a log file, and saves the log file as text
SaveAsDlf.pl	Shows an example of how to save the current file as a new .dlf file
GetField.pl	Shows how to use the FieldExists, GetField, and GetFieldArray interfaces

**NOTE:** One or more of the sample scripts assumes the presence of a log file in ~\Sample.dlf. The QCAT installation and scripts do not create or copy any log file to the Temp folder. Users need to copy a Sample.dlf log file to the specified path.

## 7.2 QCAT automation objects

QCAT 6.x provides a scripting interface that is substantially similar to the Windows QCAT 6.x COM interface.

### 7.2.1 QCAT6Application object

This is the main application object. Creating this object invokes the QCAT application. The application terminates when the script releases this object.

The following code snippet demonstrates how to create the application object.

```
use QCATDBus;
my $qcat_app = newQCAT6Application();
if(!$qcat_app)
{
print "ERROR: Unable to invoke the QCAT application.\n";
die;
}
```

The following code snippet demonstrates how to release the application object.

```
$qcat_app->Exit();
$qcat_app = NULL;
```

#### 7.2.1.1 QCAT6 properties

**bool AnnotateSILK()**

**void SetAnnotateSILK(bool)**

Values are:

- TRUE – OTA messages processed by SILK include annotations
- FALSE – SILK output is not annotated



**String AppVersion()**

This is the application version string in the format: 0n.nn.nn.

**unsigned int BandClass()****void SetBandClass(unsigned int)**

This sets the band class to use for single packet parsing.

**LogPacket\* FirstPacket(QCAT6Application\*)**

This is a packet automation object that wraps the first packet in the log file. This function is called via the QCATDBus interface, e.g.:

```
my $packet = FirstPacket($qcat_app);
if(!$packet)
{
    my $lastError = $qcat_app->LastError();
    print "Error: $lastError\n\n";
}
```

**String LastError()**

This is the description of the last error encountered.

**unsigned int LogStartTime()**

This is the start time of the log as the number of seconds since midnight Jan 1, 1970 (UTC).

**unsigned int LogEndTime()**

This is the end time of the log as the number of seconds since midnight Jan 1, 1970 (UTC).

**double LogDuration()**

This is the time span of the log file in seconds.

**unsigned int Mode()****void SetMode(unsigned int)**

This sets the timestamp mode for single packet parsing.

- 0 – UNKNOWN\_MODE
- 1 – CDMA\_MODE
- 4 – WCDMA\_MODE
- 5 – GSM\_MODE

**unsigned int Model()****void SetModel(unsigned int)**

This sets the model for single packet parsing.

**unsigned int PacketCount()**

This is the number of packets in the currently open log file.

**unsigned int VisiblePacketCount()**

This is the number of visible packets in the currently open log file. Visible packets are those that are neither filtered out nor outside of the active time window.

**PacketFilter\* PacketFilter(QCAT6Application\*)**

This is the filter automation object for removing packets by type code. This function is called via the QCATDBus interface, e.g.:

```
my $filter = PacketFilter($qcat_app);
if(!$filter)
{
    my $lastError = $qcat_app->LastError();
    print "Error: $lastError\n\n";
}
```

**PacketFilter\* EventFilter(QCAT6Application\*)**

This is the filter automation object for removing events by event ID. This function is called via the QCATDBus interface, e.g.:

```
my $filter = EventFilter($qcat_app);
if(!$filter)
{
    my $lastError = $qcat_app->LastError();
    print "Error: $lastError\n\n";
}
```

**EntityFilter\* DebugMsgFilter(QCAT6Application\*)**

This is the filter automation object for removing debug messages by component and level. This function is called via the QCATDBus interface, e.g.:

```
my $filter = DebugMsgFilter($qcat_app);
if(!$filter)
{
    my $lastError = $qcat_app->LastError();
}
```

```
    print "Error: $lastError\n\n";
}
```

### **unsigned int PRev()**

### **void setPRev(unsigned int)**

This sets the PRev to be used by SILK in decoding OTA messages.

### **unsigned int PilotInc()**

### **void SetPilotInc(unsigned int)**

This sets the Pilot Inc to be used by SILK in decoding OTA messages.

### **bool ShowHexDump()**

### **void SetShowHexDump(bool)**

Values are:

- TRUE – Print packet hex dump in addition to the pretty print
- FALSE – Do not print the hex dump

### **String SILKVersion()**

This is the SILK revision string in the format: n.nn.

### **unsigned int\* SupportedPackets()**

This returns an array of type codes. These type codes are the packet types supported by QCAT.

### **unsigned int StateRequired(unsigned int packetType)**

This returns a bitmask that identifies the log state information that must be set to correctly process a packet of the given type.

- 0x00000000 – REQUIRES\_NONE
- 0x00000001 – REQUIRES\_MODE
- 0x00000002 – REQUIRES\_P\_REV
- 0x00000004 – REQUIRES\_BANDCLASS
- 0x00000008 – REQUIRES\_MODEL

### **String TimestampFormat()**

### **void SetTimestampFormat(String)**

This sets and gets the timestamp display format; valid values are:

- Calendar – Displays all dates as calendar dates
- Days – Displays a number of days rather than a date

## **String TimestampLocale()**

### **void SetTimestampLocale(String)**

This sets and gets the timestamp locale; valid values are:

- UTC – All times will be printed as UTC time
- Local – All times will be adjusted to the local time zone

## **bool UseT53Standard()**

### **void SetUseT53Standard(bool)**

This controls use of the T53 standard for decoding OTA messages.

## **bool UsePCTime()**

### **void SetUsePCTime(bool)**

Values are:

- TRUE – Attempts to align the timestamps of the log packets with PC system time
- FALSE – Uses log packet timestamps as they are

## **bool V3xCompatibility()**

### **void SetV3xCompatibility(bool)**

Values are:

- TRUE – Uses QCAT 3.x versions of the packet names
- FALSE – Uses QCAT 4.x+ naming (recommended)

## **bool PrefixLogName()**

### **void SetPrefixLogName(bool)**

Values are:

- TRUE – Prefix log name to outputs
- FALSE – Not prefix log name to outputs

## **Workspace\* Workspace(QCAT6Application\*)**

This is the workspace object. This function is called via the QCATDBus interface, e.g.:

```
my $workspace = Workspace($qcat_app);
if (!$workspace)
{
    my $lastError = $qcat_app->LastError();
    print "Error: $lastError\n\n";
}
```

### 7.2.1.2 Methods

#### **void Force3GPPRev**

This sets the 3GPP revision for parsing WCDMA RRC Signaling messages. This is necessary only when the correct 3GPP revision cannot be determined from the log file.

```
(  
    unsigned int nMonth,  
    unsigned int nYear  
);
```

#### **bool GenerateVocoderPCM**

This extracts vocoder frames from log packets, concatenates, and converts them to a PCM file that can be played back using a PCM audio utility.

```
(  
    String txOutPath,  
    String rxOutPath,  
    unsigned int mode,  
);
```

Parameter descriptions are:

- txOutPath – Output file path for the Tx stream
- rxOutPath – Output file path for the Rx stream
- Mode – Vocoder mode if not contained in the log packets; must be one of:
  - 0 – AUTOMATIC\_CODEEC (determine from log file, if possible)
  - 1 – AMR\_CODEEC
  - 2 – EFR\_CODEEC
  - 3 – FR\_CODEEC
  - 4 – HR\_CODEEC

## bool ExtractVocoderDataFiles

This extracts vocoder frames from vocoder packets per codec (\*.in), then converts them to a chunk of PCM (\*.out) and concatenates all chunks into a final PCM file at the end. All generated files are dumped at the outPath directory.

- \*.voc.<dir>.frame.< number>.<codec>.<dir>.in – Concatenated frames per codec
- \*. voc.<dir>.frame.< number>.<codec>.<dir>.out – .in file converted to a PCM chunk
- \*.voc.<dir>.raw – Final PCM file that is a concatenation of all .out files

```
(  
    String outPath,  
    unsigned int options,  
    unsigned int codec  
)
```

## Variant GetProperty

This gets a configuration property by name. This returns empty if the property was not found.

```
(  
    String name  
);
```

## String GetPacketTypeName

```
(  
    unsigned int type  
);
```

Parameter description is:

- Type – The packet type code to look up; returns the English name for the type

## String GetPropertyList

This returns a list of all the configuration properties in the application. These properties can be set and gotten using PutProperty and GetProperty.

```
(
    bool bIncludeValues
);
```

Parameter description is:

- bIncludeValues – Includes the current value for the property in the returned string

## bool LoadWorkspace

This returns true if the workspace was successfully opened.

```
(
    String Name
);
```

Parameter description is:

- Name – Name of the workspace to open

## bool OpenLog

This opens a log file.

```
(
    String pathName
);
```

Parameter description is:

- pathName – Full path to the log file to be opened

This returns true on success. If the result is false, it retrieves the last error message from the LastError property.

## **bool closeFile()**

This closes the log file.

## **bool Process**

This opens a log file and pretty prints the packets into an ASCII text file.

```
(
    String logPath,
    String outputPath,
    bool bHexDump,
    bool b3xCompatibility
);
```

Parameter descriptions are:

- logPath – Full path to the target log file
- outputPath – Full path of the text file to produce
- bHexDump – If true, the hex dump for each packet is printed in the output file
- b3xCompatibility – If true, the QCAT 3.x version of the packet name is used (false is recommended for new scripts)

## **bool ProcessDebug**

This opens a log file and pretty prints only the debug message packets into an ASCII text file.

```
(
    String logPath,
    String outputPath
);
```

Parameter descriptions are:

- logPath – Full path to the target log file
- outputPath – Full path of the text file to produce

## **String ParsePackets**

```
(
    unsigned int start,
    unsigned int count
);
```

Parses a range of packets within the current file into text.



Descriptions are:

- start – First packet to parse to text
- count – Total number of packets to parse to text (begging with start)

## **bool PutProperty**

This sets a configuration property by name. Valid configuration property names can be listed using the GetPropertyList method.

```
(
    String name,
    Variant value
);
```

Parameter descriptions are:

- Name – Name of the property to be set
- Value – New value for the property

This returns false if the property with the given name is not found.

## **bool SaveAsText**

This parses the current log file and saves the output text to the file specified by pszOutPath.

```
(
    String outPath
);
```

Parameter description is:

- outPath – Output file path

## **bool SaveAsDLF**

This saves the current log file as a new .dlf file.

```
(
    String outPath
);
```

Parameter description is:

- outPath – Output file path

## bool SaveAsUpdatedDLF

This saves the current log file as a new .dlf file, updating the timestamps to be as they are displayed, which means saving the PC time offset into the log headers.

```
(  
    String outPath  
);
```

Parameter description is:

- outPath – Output file path

## bool SaveAsISF

This saves the current log file as a new .isf file.

```
(  
    String outPath  
);
```

Parameter description is:

- outPath – Output file path

## bool SaveAsUpdatedISF

This saves the current log file as a new .isf file, updating the timestamps to be as they are displayed, which means saving the PC time offset into the log headers.

```
(  
    String outPath  
);
```

Parameter description is:

- outPath – Output file path

## void SetTimeWindow

This limits packets to a specified time window.

```
(  
    double offset,  
    double length  
);
```

Parameter descriptions are:

- Offset – Beginning of the window as an offset from the first packet in the log (in seconds)
- Length – Size of the window in seconds

### **void SetTimeWindowAbsolute**

This limits packets to a specified time window.

```
(
    String startTime,
    String endTime
);
```

Parameter descriptions are:

- startTime – A string containing the time at which to start the time window
- endTime – A string containing the time at which to end the time window

The string for the startTime and endTime is formatted as yyyy/mm/dd hh:mm:ss.ddd, e.g., 2013/01/05 12:05:23.123.

### **void SetPacketWindow**

This limits packets to a specified window.

```
(
    unsigned int offset,
    unsigned int length
);
```

Parameter descriptions are:

- Offset – Number of packets to be clipped off the front of the log
- Length – Number of packets to be viewed

### **bool SortByTime();**

This sorts the packets of the open log file by timestamp.

### **bool SortByIndex();**

This sorts the packets of the open log file by file index.

### **bool SortByLogId();**

This sorts the packets of the open log file by log ID.

### **bool SortByLogName();**

This sorts the packets of the open log file by log name.

## **bool SortBySize();**

This sorts the packets of the open log file by log size.

## **7.2.2 LogPacket**

### **7.2.2.1 Properties**

#### **unsigned int Type()**

This is the packet type code.

#### **unsigned int Length()**

This is the length of the packet in bytes (including DLF item header – 12 bytes).

#### **String Name()**

This is the name for this type of packet.

#### **String TimestampAsString()**

This is the timestamp as a date/time string.

#### **String Text()**

This is the parsed payload of the packet as would appear in the QCAT text output.

#### **unsigned int SortedIndex()**

This is the time-sorted index of the packet. It can be used in conjunction with functions taking a packet index, e.g., SetPacketWindow.

#### **String Subtitle()**

This is the subtitle of this packet.

- Events – This is the event name
- Debug messages – This is the message name
- OTA packets – This contains information about the payload

### **7.2.2.2 Methods**

#### **bool Next();**

Go to the next packet.

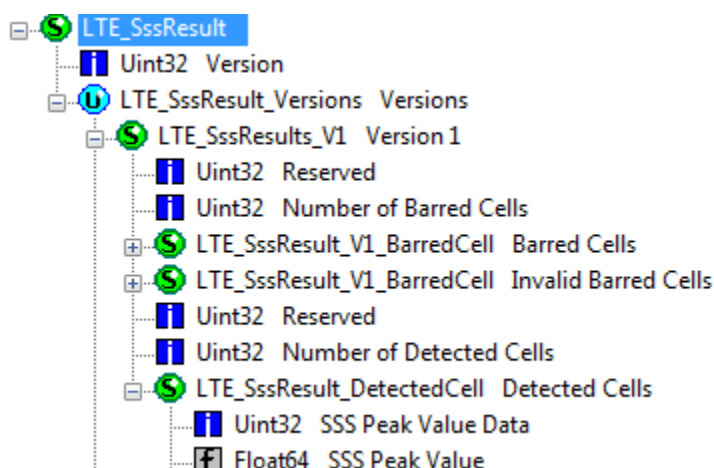
This returns false if this is the last packet.

#### **bool FieldExists();**

This returns true when a string specifier references a valid field within the packet. The string specifier is described in GetField().

## String GetField();

This returns the first instance of data where the field name matches the string specifier within the packet. The string specifier is a portion of the log structure definition, where as much or as little of the structure can be specified as desired, with as little as the field name required. It can also specify indexes in arrays. For instance, given the following structure:



Example accessors are:

Accessor	Description
."SSS Peak Value"	Matches every instance of SSS Peak Value in the packet
."SSS Peak Value"[0]	Matches only instances of SSS Peak Value at array index 0
."Detected Cells"."SSS Peak Value"	Matches SSS Peak Value within the Detected Cells field
."Detected Cells"[1]."SSS Peak Value"	Matches SSS Peak Value within the Detected Cells field, where Detected Cells is at index 1

## double[] GetFieldArray();

This returns every instance of data where the field name matches the string specifier within the packet. The string specifier is described in GetField(). If the value to be returned cannot be cased into a double, i.e., strings, then it is skipped.

## 7.2.3 PacketFilter

### 7.2.3.1 Methods

#### **void SetAll**

This enables/disables all packet types.

```
(
    bool bEnable
);
```

Parameter description is:

- bEnable
  - TRUE – Enable
  - FALSE – Disable

#### **Void SetRange**

This enables/disables a range of packet types (inclusive).

```
(
    unsigned int nFirst,
    unsigned int nLast,
    bool bEnable
);
```

Parameter descriptions are:

- nFirst – First packet type to be enabled/disabled
- nLast – Last packet type to be enabled/disabled
- bEnable
  - TRUE – Enable
  - FALSE – Disable

#### **void Set**

This enables/disables a single packet type.

```
(
    unsigned int type,
    bool bEnable
);
```

Parameter description is:

- type – Type of packet to disable
- bEnable
  - TRUE – Enable
  - FALSE – Disable

### **bool IsEnabled**

This returns true if the given packet type is enabled.

```
(  
    unsigned int type  
);
```

Parameter description is:

- type – Packet type to query

### **void Commit();**

Apply the current filter to the log file.

## **7.2.4 Workspace**

### **7.2.4.1 Methods**

#### **bool ExportToText**

This runs all analyses selected in the workspace and stores the results in text files with data columns delimited by CellDelimiter (usually the tab character “\t”).

```
(  
    String strOutputPath,  
    String strCellDelimiter  
);
```

OutputPath is the absolute path name.

#### **bool ExportToBitmap**

This runs all analyzers selected in the workspace and stores the resulting graphs in .png files.

```
(  
    String strOutputPath  
);
```

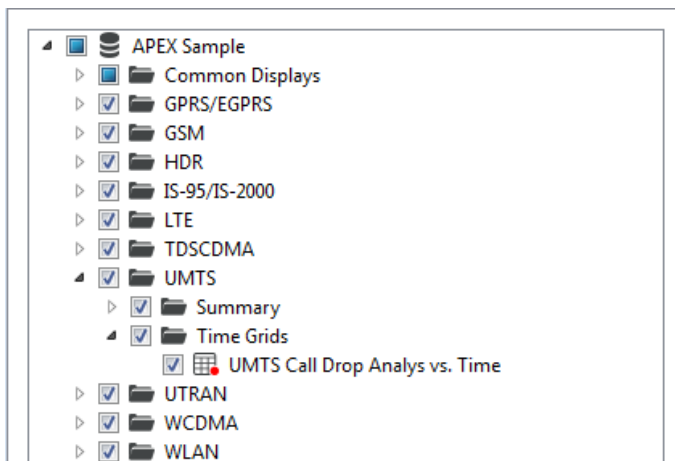
OutputPath is the absolute path name.

## bool SelectOutput

This selects a display or a folder of displays (by path) in the workspace to be exported.

```
(
    String strPath,
    String bEnable
);
```

To select a workspace element, concatenate the names of all parent folders with the item's name separated by semicolons. A lone semicolon represents the root. [Figure 7-1](#) shows an examples path for the SelectOutput method.



**Figure 7-1 Example for a path for the SelectOutput method**

For example, the path to UMTS Call Drop would be:

“;UMTS;Time Grid;UMTS Call Drop Analysis vs. Time”



**unsigned int\* GetRequiredPackets**

This returns an array of the log codes used by the specified analyzer.

```
(  
    String strPath  
);
```

The specifier is the same as used in SelectOutput.

**unsigned int\* GetRequiredEvents**

This returns an array of event IDs used by the specified analyzer.

```
(  
    String strPath  
);
```

The specifier is the same as used in SelectOutput.

## 8 QCAT Excel Workbook (Windows Only)

---

To generate the Excel workbook from one or more of the QCAT displays, first export those displays to a spreadsheet format before further use. For more information about exporting QCAT displays, see Section [3.6.29.11.2](#).

Features of the QCAT Excel workbook are:

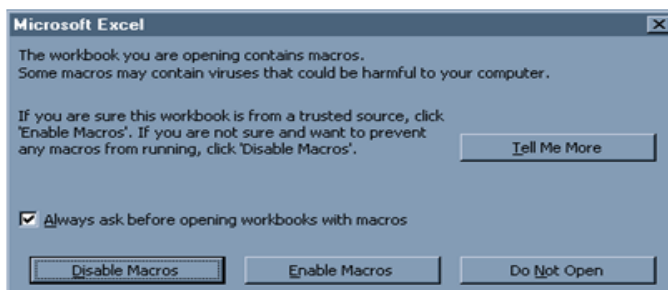
- Table of contents provided as the first worksheet
- QCAT navigator toolbar that provides:
  - Easy navigation to and from any worksheet
  - Zooming capability
  - Study functionality – The ability to combine data from different vs. Time Plots with a greater level of detail for study purposes

### 8.1 Viewing the QCAT workbook

#### 8.1.1 Getting started

Generated Excel workbooks are listed in the bottom view in the Output Files tab. To view the contents of the Excel file, double-click the filename. This prompts the respective applications to invoke Excel to open the given workbook. The workbook can also be invoked from My Computer or Windows Explorer like any other Excel file. The workbook must be opened with a version of Excel that is the same or later than the version used to create it. The export engine uses OLE automation constructs that are compatible with Microsoft Excel 2007 and above.

It is important that Excel has the Enable Macros setting enabled. If this is not already enabled, a dialog to disable or enable macros will display. If such a dialog is displayed, select Enable Macros. The QCAT navigator (see Section [8.1.4](#) for details) is not available if macros are not enabled.

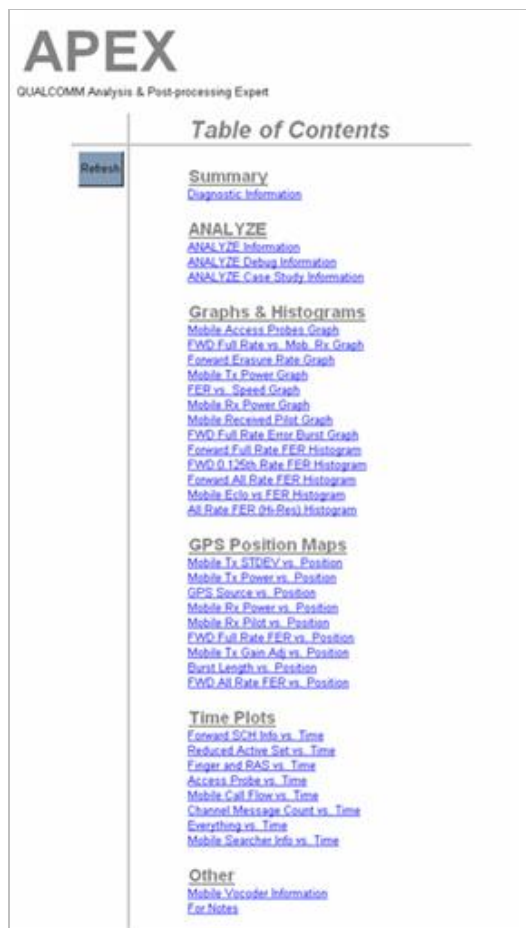


## 8.1.2 QCAT workbook index page

The QCAT workbook index page is the first sheet in the QCAT workbook and is displayed first when a new workbook is opened. This sheet contains the table of contents for the entire workbook; a sample is shown below. It lists all the visible worksheets in the workbook and provides hyperlinks to each sheet. All sheets in the workbook can be accessed from the table of contents.

The entries in the table of contents are classified into the following categories:

- Summary/Analyze sheets – ASCII data that may not fit any fixed tabular format
- Graphs and Histograms – Data is presented alongside the formatted graph or histogram
- GPS Position Maps – Maps data by GPS location of the UE
- Time Plots – Large tables of data listed against timestamps
- Summary – Sheets that do not fit the above classification



**APEX**  
QUALCOMM Analysis & Post-processing Expert

**Table of Contents**

[Refresh](#)

**Summary**  
[Diagnostic Information](#)

**ANALYZE**  
[ANALYZE Information](#)  
[ANALYZE Debug Information](#)  
[ANALYZE Case Study Information](#)

**Graphs & Histograms**  
[Mobile Access Probes Graph](#)  
[FWD Full Rate vs. Mob. Rx Graph](#)  
[Forward Erasure Rate Graph](#)  
[Mobile Tx Power Graph](#)  
[FER vs. Speed Graph](#)  
[Mobile Rx Power Graph](#)  
[Mobile Received Pilot Graph](#)  
[FWD Full Rate Error Burst Graph](#)  
[Forward Full Rate FER Histogram](#)  
[FWD 0.125s Rate FER Histogram](#)  
[Forward All Rate FER Histogram](#)  
[Mobile Echo vs FER Histogram](#)  
[All Rate FER \(H-Best\) Histogram](#)

**GPS Position Maps**  
[Mobile Tx STDEV vs. Position](#)  
[Mobile Tx Power vs. Position](#)  
[GPS Source vs. Position](#)  
[Mobile Rx Power vs. Position](#)  
[Mobile Rx Pilot vs. Position](#)  
[FWD Full Rate FER vs. Position](#)  
[Mobile Tx Gain Adj vs. Position](#)  
[Burst Length vs. Position](#)  
[FWD All Rate FER vs. Position](#)

**Time Plots**  
[Forward SCH Info vs. Time](#)  
[Reduced Active Set vs. Time](#)  
[Finger and RAS vs. Time](#)  
[Access Probe vs. Time](#)  
[Mobile Call Flow vs. Time](#)  
[Channel Message Count vs. Time](#)  
[Everything vs. Time](#)  
[Mobile Searcher Info vs. Time](#)

**Other**  
[Mobile Yacoder Information](#)  
[For Notes](#)

The table of contents sheet also has the Refresh button which redraws the entire table. This button can be used to update the table of contents actions performed, e.g., adding a custom sheet to the workbook, or hiding, unhiding, or deleting an existing sheet from the workbook. The workbook does not keep track of such events automatically.

### 8.1.3 QCAT workbook study sheet

A study sheet can be added to the existing workbook to analyze and combine data from different vs. time plots and grids. To create a study sheet, click **Create Study Sheet** from the study drop-down list in the QCAT navigator toolbar. One of the XY scatter chart sheets should be activated first.

The QCAT-generated workbook provides the ability to:

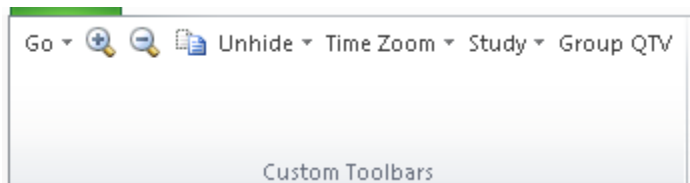
- Combine any vs. time plots
- Zoom to a greater level of detail

Add events, debug messages, and TPC rejected bits on the created study sheet; for details, see [Table 8-1](#).

The time zooming functionality works on any vs. time plot worksheet in the workbook.


### 8.1.4 QCAT navigator

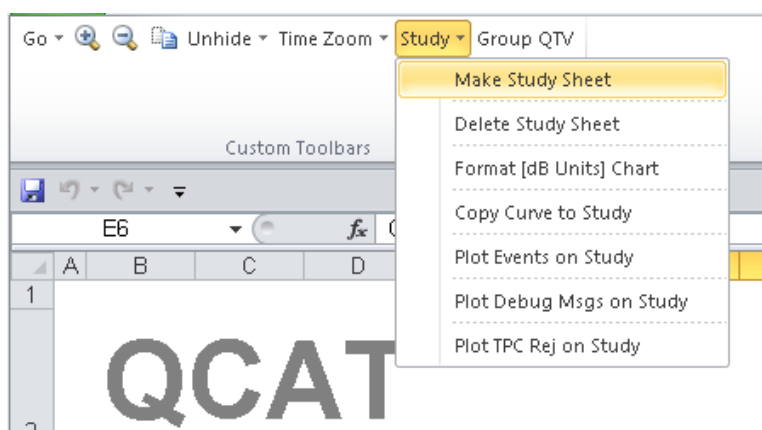
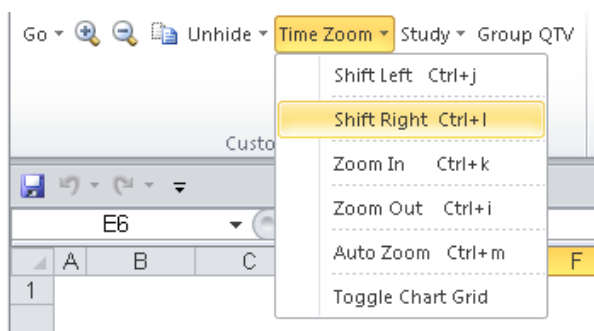
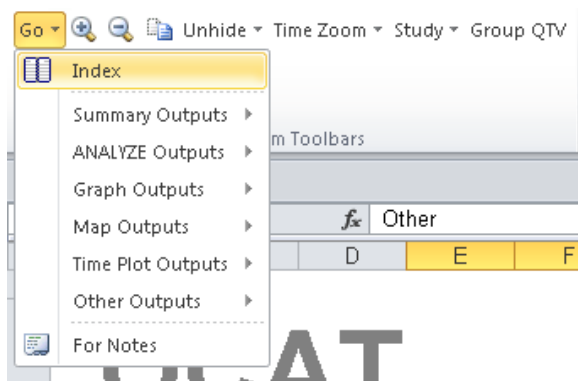
The QCAT navigator is a custom Excel toolbar created by QCAT for use with the QCAT workbook. QCAT navigator functions are described in [Table 8-1](#). To retrieve the toolbar, select the Add-Ins tab.



**Table 8-1 QCAT navigator functions**

Function name	Description
Go ▼	Contains a pull-down list of all visible sheets in the workbook; allows navigation from one sheet to another
Zoom in 🔍	Zooms in on the active worksheet
Zoom out 🔍	Zooms out on the active worksheet
Hide ▼	Hides the selected sheet from the active list
Unhide ▼	Unhides the selected sheet
Time Zoom ▼	Contains a pull-down list of commands that allows manipulation of the XY scatter plots to a greater level of detail The commands are: <ul style="list-style-type: none"> <li>■ Shift left (Ctrl+J) – Moves plot to the left</li> <li>■ Shift right (Ctrl+L) – Moves plot to the right</li> <li>■ Zoom in (Ctrl+K) – Zooms in</li> <li>■ Zoom out (Ctrl+I) – Zooms out</li> <li>■ Auto Zoom (Ctrl+M) – Returns to a default scale</li> <li>■ Toggle Chart Grid – Turns the chart grid on or off on the plot</li> </ul>

Function name	Description
Study 	<p>Contains a pull-down list of commands that allows the creation of a copy (study sheet) of an active vs. time plot sheet to manipulate/enhance it. There is no undo function for the commands; delete the study sheet and start again if necessary. To create a second study sheet, rename the current one. All Study commands, except format, will not work on user-renamed study sheets. Clicking Make Study Sheet when one already exists will make the current study sheet active. The commands are:</p> <ul style="list-style-type: none"> <li>▪ Make Study Sheet – Creates a study sheet (copy) from any active vs. time plot sheet</li> <li>▪ Delete Study Sheet – Deletes an existing study sheet</li> <li>▪ Format [dB Units] Chart – Performs special formatting (requires the chart to have dB units on the primary Y-axis). Works on both study sheets and vs. time plot sheets. Adds a grid, performs some scaling on the primary Y-axis, removes the previous chart title, and changes the primary Y-axis units name to Power [dB].</li> <li>▪ Copy Curve to Study – Copies the selected curve to the study sheet. The curve must be of type XYScatter. The Y-axis will be rescaled during the copy and will retain its color unless the user changes it manually. To remove the copied curve from the study sheet, select and delete it.</li> <li>▪ Plot Events on Study – This plots events onto the study sheet. Y-values are plotted as Event IDs scaled by 1000 on the secondary Y-axis. An events vs. time sheet must be present. The macro turns on Excel's autofilter for the events vs. time sheet and plots all of the events onto the study sheet. Thereafter, select different sets of events from the events vs. timesheet using autofilter, and they are correspondingly plotted onto the study sheet.</li> <li>▪ Plot Debug Msgs on Study – This plots debug messages onto the study sheet. Y-values are plotted as 0 on the secondary Y-axis. A debug messages vs. time sheet must be present. The macro turns on Excel's autofilter for the debug messages vs. time sheet and plots all of the events onto the study sheet. Thereafter, select different sets of debug messages from the vs. time sheet using autofilter, and they are correspondingly plotted onto the study sheet.</li> <li>▪ Plot TPC Rej on Study – This plots the number of TPC rejections for the first three combiners onto the study sheet. A WCDMA TPC history vs. time sheet must be present, and the number of TPC rejection column must have data in it. Three additional columns are also added to the WCDMA TPC history vs. time sheet if all necessary data is present.</li> </ul>



# A References

---

## A.1 Related documents

Document	
<b>Qualcomm Technologies, Inc.</b>	
<i>PC Tool DLF Format Specification</i>	80-V1595-1
<i>CDMA Dual-Mode Subscriber Station Serial Data Interface Control Document</i>	80-V1294-1
<i>Serial Interface Control Document for WCDMA</i>	80-V2708-3
<i>Serial CD for GSM, GPRS, and EGPRS Interface Control Document</i>	80-V5295-1
<i>Serial Interface Control Document (ICD) for UMTS</i>	80-V4083-1

## A.2 Acronyms and terms

Term	Definition
CB	Cell broadcast (layer 3 tTask)
DS	Data services (task)
EVS	Enhanced voice service
GL1	GSM/GPRS layer 1
GL2	GSM/GPRS layer 2
GHDI	Generic hardware-driver interface
GSDI	Generic SIM-driver interface
MM	Mobility management (layer 3 task for GSM)
MN	Mobile network layer (layer 3 task)
RLC DL	Radio link control downlink (layer 2 task for WCDMA or GPRS)
RLC UL	Radio link control uplink (layer 2 task for WCDMA or GPRS)
RR	Radio resource (layer 3 task for GSM)
SACCH	Standalone control channel
SM	Session management
SNDC	Subnet dependent converge protocol (GRPS)
SS	Supplementary service
WL1	WCDMA layer 1
WL2	WCDMA layer 2