TELEDYNE SP DEVICES
Everywhereyoulook™

ADQAssist

User Guide

Author(s): Teledyne SP Devices

Document ID: 20-2521
Classification: Public
Revision: 1
Date: 2023-05-03

Contents

1 Introduction

1.1 About ADQAssist

ADQAssist is a GUI based maintenance tool for ADQ digitizers. Its features include, but not limited to, support for firmware update, log tracing/viewing and license management. ADQAssist simplifies all these tasks by utilizing different necessary tools under the hood while engaging the user interaction via a single umbrella user-interface. This alleviates the users from some of the complexities of having to manage different tools for different tasks. ADQAssist cannot be used to directly capture data from ADQ devices. For that and other non-maintenance related operations, please refer to Digitizer Studio instead.

1.2 About this document

The purpose of this document is to provide a basic usage guide for ADQAssist to handle the most common everyday maintenance tasks for ADQ devices. Starting out with some minor details on how ADQAssist actually works and its dependency on other applications and moving on to some basic application navigation. Lastly but most importantly, a few practical common usage scenarios will be outlined and explained. This documentation is limited to the core functionalities of ADQAssist which are intended for the end-users of ADQ devices. Features intended for debugging purposes by TSPD support will not be covered here. Some useful data visualization feature that might come in handy for first time digitizer users will also be mentioned breifly.

1.3 Operating system support

ADQAssist is available for both Windows and Linux. The Linux support is limited to the distributions listed in Table 3. Apart from some differences in the installation method and structure, most but not all features and functionalities are identical on both Windows and Linux.

1.4 Definitions and Abbreviations

Table 1: Definitions and abbreviations used in this document.

Item	Description
TSPD	Teledyne SP-Devices
ADQAPI	ADQ Application Programming Interface
GUI	Graphical User Interface
View/Panel	A subsection or area of the main application window.
User-Application	Software program that uses ADQAPI to collect data from an ADQ device
Call-Script	A programming script used to setup an enviorment for an application and to start it

2 Installation

2.1 Windows

ADQAssist is normally bundled within the SDK installer on Windows. The application shortcut can be found on the Windows Start Menu just like ADCapturelab and Digitizer Studio. The application file itself is located in *C:/Program Files/SP Devices/ADQUpdater* by default. The Windows version of ADQAssist is a self-contained portable executable file that can be moved to other location if desired. However, doing so might disable some of the features related to other sub-programes that ADQAssist needs access to.

2.2 Linux

On Linux, ADQAssist is distributed via in the *ADQGuiTools* package, together with Digitizer Studio. This Linux package is mainly aimed towards GUI based usage of ADQ Devices. Also on Linux, the application can be started via the Start Menu shortcut under the *Utilities* category. The ADQAssist application file itself is installed into the */opt/adqguitools/bin* location. Due to the nature of Linux and its different flavours, the ADQAssist executable file is not portable as it is on Windows and should not be relocated carelessly. It also can only be started properly via the included call-script (*adqassist.sh*) or thru the desktop shortcut. If relocation is still desired for testing purposes, the whole *adqguitools* folder should be moved instead to keep the installation structure intact. This way, the relocated version of ADQAssist can still be started via its call-script within its moved folder structure.

3 Features and external tools dependency

ADQAssist has two sets of features. The first set of features is completely independent from any other external tools and the other set will be depending on the existence of the required external tools. Even without the external tools, ADQAssist can still be used stand-alone with its built-in features. Only when accessing the features from the external tools will it check for the existence of these tools. The location of these tools should either be the default installation path or the same folder that the ADQAssist executable file itself resides in. If ADQAssist has been relocated to some other place, make sure it can find the necessary tools if you want to utilize the features from these tools. See Table 2 below for more details on which tool is required for what type of operations. Some feature such as the log monitoring can be utilized both via the built-in monitor or via the external tool.

Table 2: Features and tool dependncy

Feature Group	Tool Dependency	Tool Name
Log Monitor/View	Built-in/External	ADQUpdater, ADQUpdater_Legacy
Data Visualization	Built-in	
Firmware Operations	External	ADQUpdater, ADQUpdater_Legacy
License Operations	External	ADQLicenseUtil

ADQAssist also makes an effort to inform the user what commands are being sent to the external com-

mand line tools by printing out the command string in the *Console* view. This can be useful if the user intends to directly use the command line tools when occasion requires.

Note

Command line strings sent to the external tools are displayed in the *Console* view in blue color. They can be useful to directly call the external tools via a terminal.

4 Application Navigation

ADQAssist employs a simple but dynamic GUI layout, which sometimes can make it a little tricky for new users to get used to. The basic design philosophy is to offer personalized layout customization depending on usage scenario and effective utilization of screen real estate. Views that are not needed can be tucked away and visible views can be rearranged to fit personal preferences.

There are 3 distinctive visual components that make up the whole ADQAssist GUI:

1.) The menu row on the top
2.) The toolbar row (re-arrangeable)
3.) The different views that serve different features (re-arrangeable)

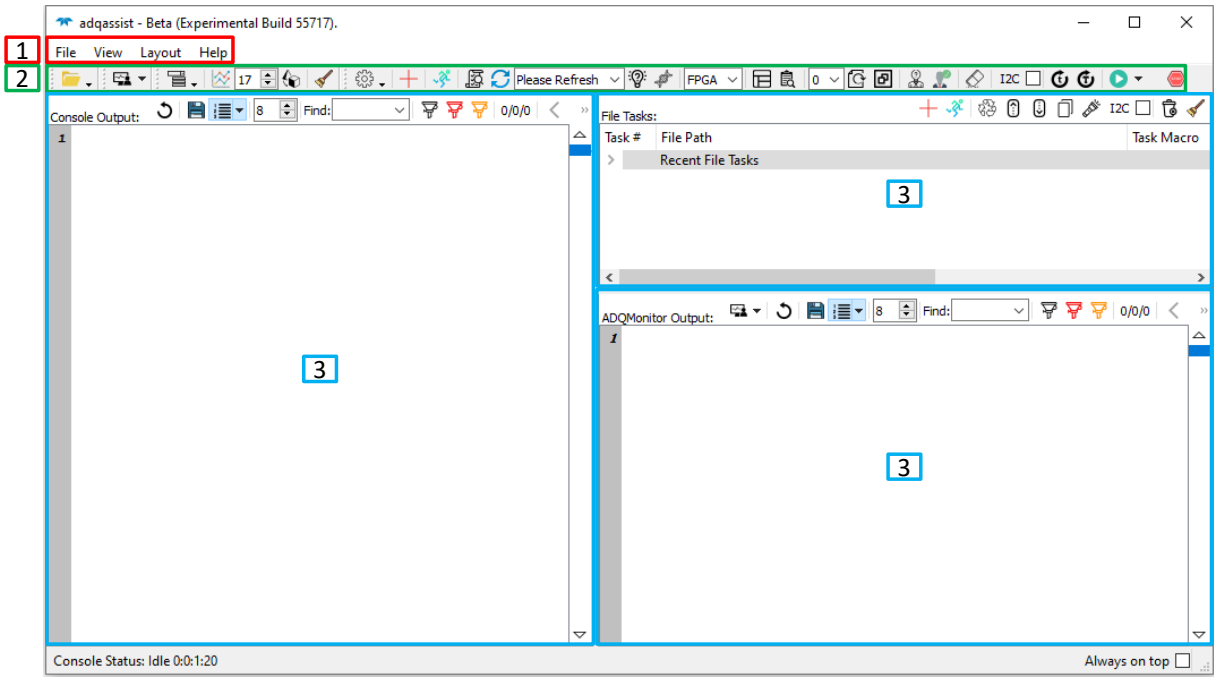

Figure 1: Main visual components of ADQAssist

Within each view there might also be some fixed toolbar row to aid the specific features offered by that view. Some features that can be found in the main menu row are also accessable via the right-click context menu for each view. These features are often related to the view's visibility and layout. The context menu can be triggered by right-clicking on the empty space at the top edge of each view. If the view titlebar is *not visible*, double-clicking on this empty space will maximize the view and occupy all the space within the main window. If the view titlebar *is visible*, double-click on the titlebar instead to get the same result.

Note

Right-click on the empty space at the top of a view to access context menu. Doucle-clicking on this area will maximize the current view (*If the view titlebar is not visible*).

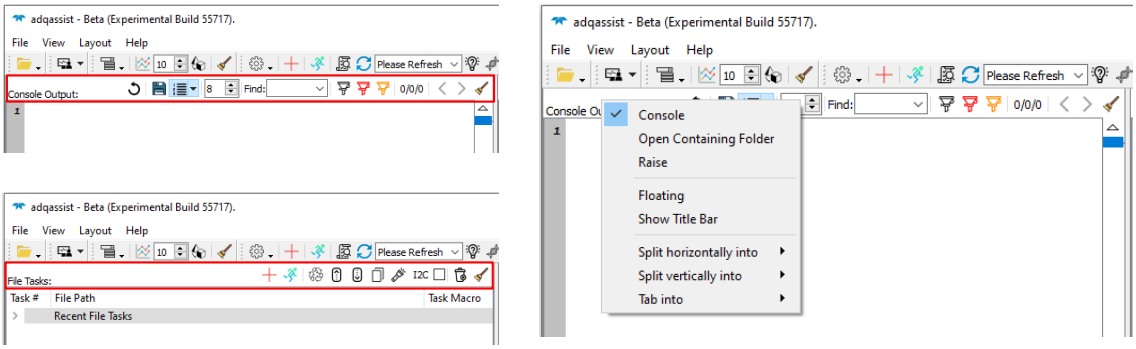

Figure 2: Fixed toolbars and right-click context menu on the empty space at the top of each view.

Whenever possible, every GUI control element of ADQAssist is equiped with a tooltips which breifly describes what that control is used for. This is particularly useful for new users. To see the tooltips, just hover the mouse pointer over the control element of interest.

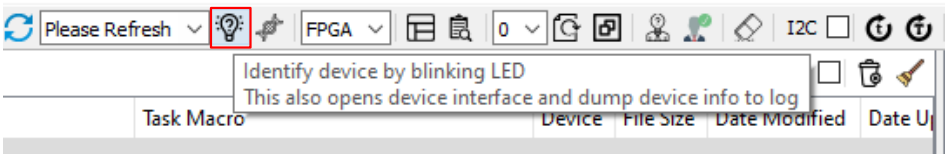

Figure 3: Every button has a tooltips that breifly describes its functionality.

Note

Hover mouse pointer over a toolbar button to get useful information about what that button does.

4.1 Views and Presets

There are in total 6 different dockable views that can be rearranged into different layout configurations depending on the usage scenario. A dockable view is a subwindow that can be teared off into its own separate window. This is particularly useful if the user wants to utilize multiple screens, each for different view. Some of the views are related to each other and thus will be shown automatically if an operation from one view triggers a change in a related view. For instance the *Console* view will be shown whenever a task is initiated in the *File Tasks* view.

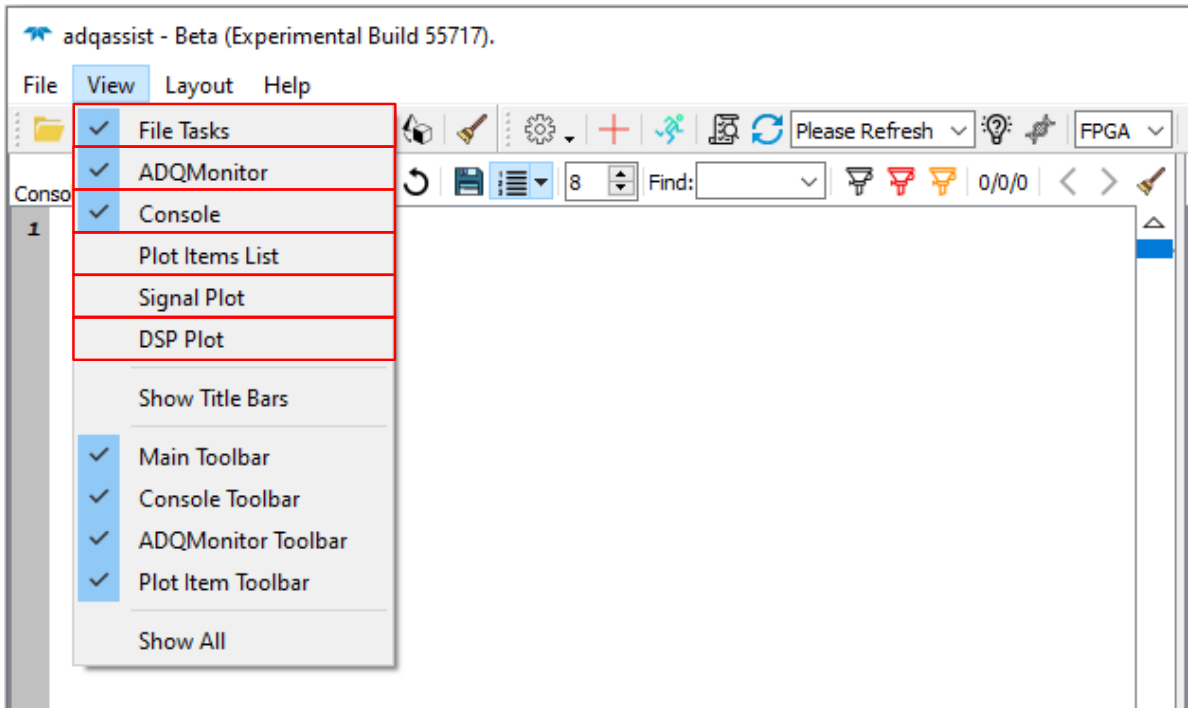

Figure 4: The visibility for the 6 different views can be swtiched on or off via the View menu

Rearrangment of each view can be done via the *Layout* menu, the right-click context menu or by manually dragging the titlebar of each view into the desired location of the main window. By default, the titlebar for each view is hidden to conserve vertical space of the screen. But they can easily be turned on via the *View* menu or via the right-click context menu for a particular view.

Note

Titlebar can be used to drag a view into the desired dock location, or to *undock* the view from the main window completely. Undocking is also called *Floating*.

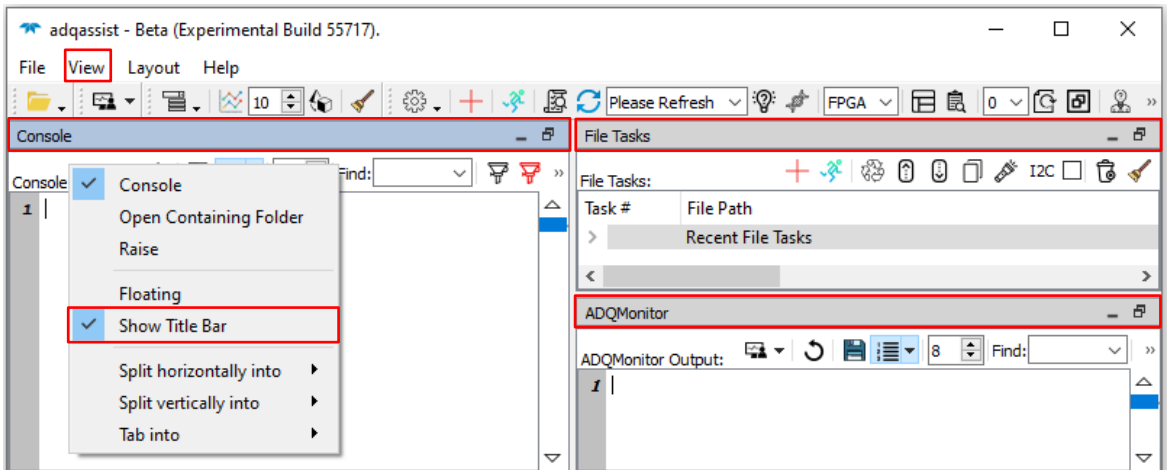

Figure 5: Titlebar can be enabled via the right-click context menu or via the *View* menu

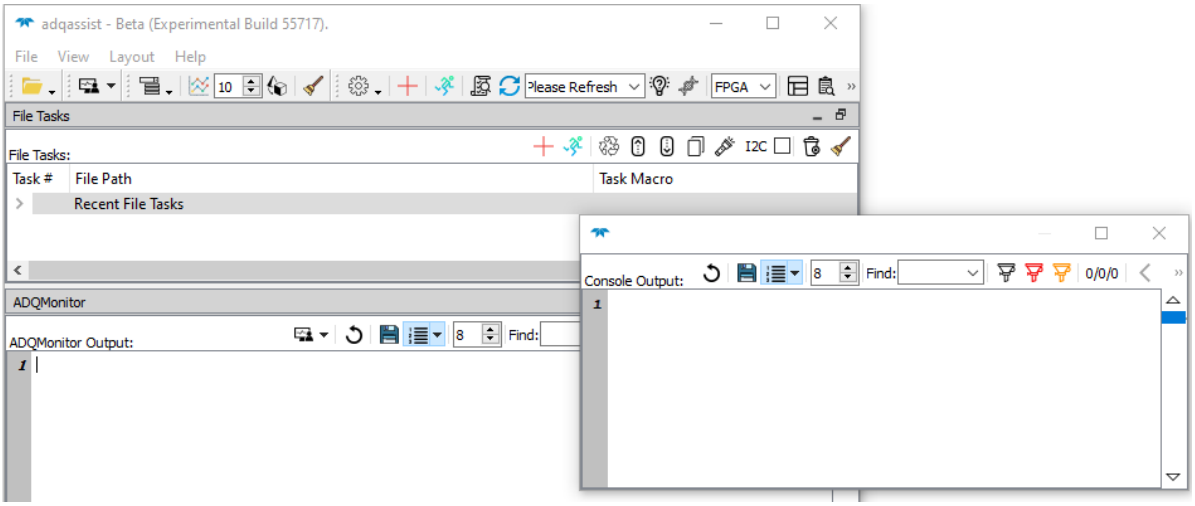

Figure 6: Each view can be teared off into its own window (*undocked*). Here showing *Console* view being undocked.

4.2 Preset Layout

There are 5 different preset layouts for the views which can be used to restore ADQAssist to a certain predetermined configuration, in case the views have been relocated too much and are out of recognition for the user. These preset layouts can be accessed via the *Layout/ Restore Layout* menu. The *Startup* preset is worth a special mention since this preset will restore the GUI into whatever configuration it was started with at the beginning of the usage session. The other presets are meant to accommodate different usage purposes based on TSPD's own estimate.

Note

F3 and F4 can be used to quickly cycle around different preset layouts.

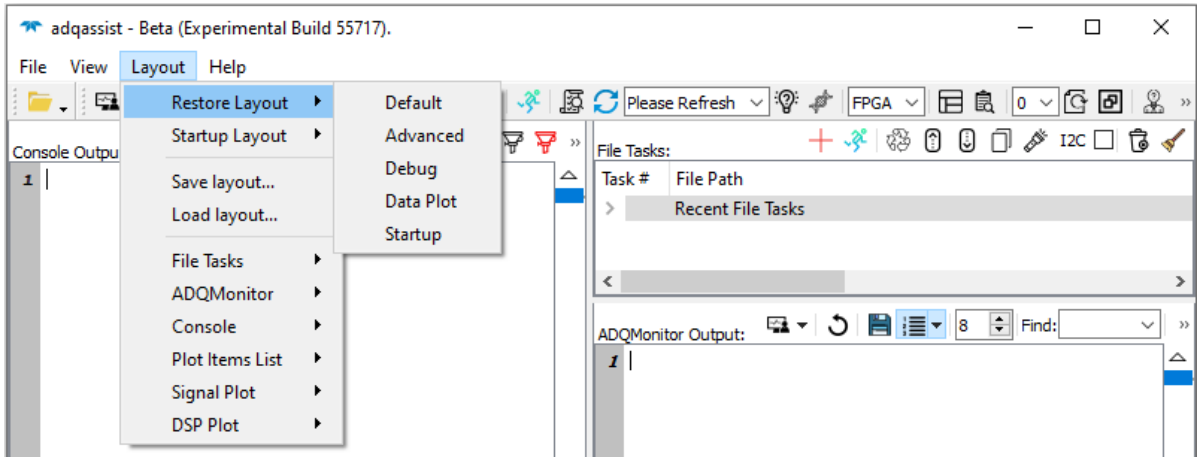

Figure 7: The default preset layouts can be used to return the GUI to a known configuration.

4.3 Startup Layout

ADQAssist can be configured to startup in 2 different ways. Either in the default and most commonly used layout configuration, or in the *Last Used* configuration before the application was shutdown. The *Last Used* layout configuration is particularly useful if the user wants to continue using the same layout from the previous session. For example when the user has manually placed the views in a particular configuration and wants to statup with this same custom layout for the next session.

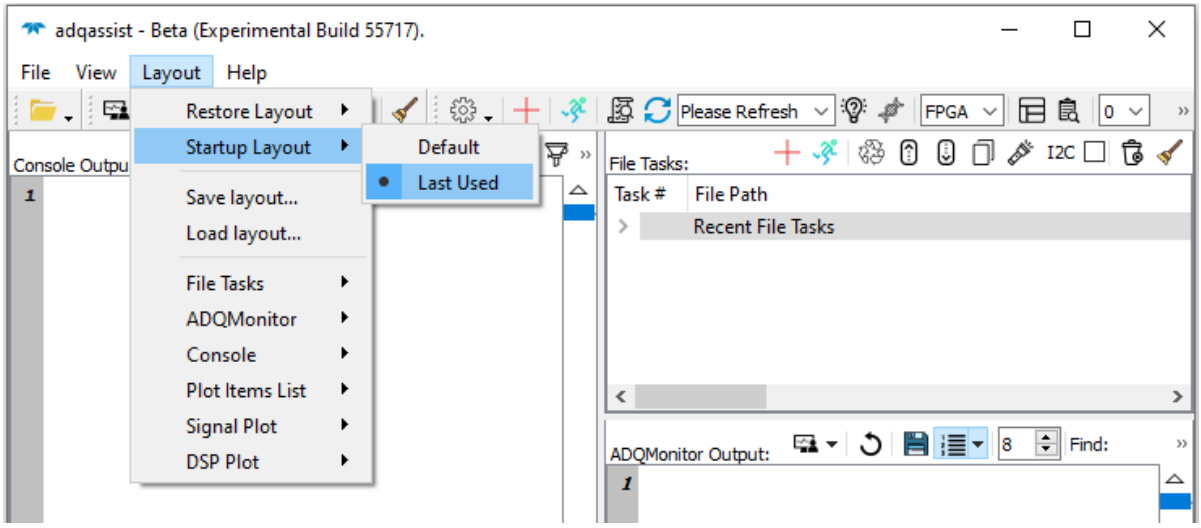

Figure 8: The *Last Used* startup layout option is particularly useful to continue from previous session.

5 Basic ADQ maintenance operations

Maintenance operations on ADQ devices performed via ADQAssist can be divided into 2 types. *Task based* operations, which can be scheduled for multiple devices at once, or *single device* based operations, which often involves in gathering and listing information of different kinds from a specific ADQ device. Most maintenance operations should be accommodated by having the ADQMonitor feature enabled to detect possible internal issues while performing the operations in question.

Note

Enabling the ADQMonitor during maintenance operations can help capturing possible internal issues.

5.1 ADQMonitor

ADQMonitor is a monitoring process that captures all log outputs from any and all applications on the entire operating system that communicates via the ADQAPI, regardless whether the running applications have enabled the log tracing or not. The outputs captured by ADQMonitor are very useful not only for TSPD staff to resolve possible support issues, but they can also be very handy for the end-users to figure out where things might have gone wrong. ADQMonitor should therefore be the most frequently used feature of ADQAssist for any ADQ users.

ADQMonitor is built into both ADQAssist and the command line application *ADQUpdater_legacy*. When using this log monitoring feature, the user has the option to chose whether to run the log monitor thru a separate external process (I.e running *ADQUpdater_legacy* in the background and redirect the output to ADQAssist) or using the built-in version via an internal thread. The differences between these two choices are related to the application's performance and the backward compatibility with older versions of ADQAPI and *ADQUpdater_legacy*. In particular for those older ADQAPI versions before ADQAssist exists. ADQMonitor can be enabled by toggling the monitor button from the main toolbar or from the ADQMonitor view.

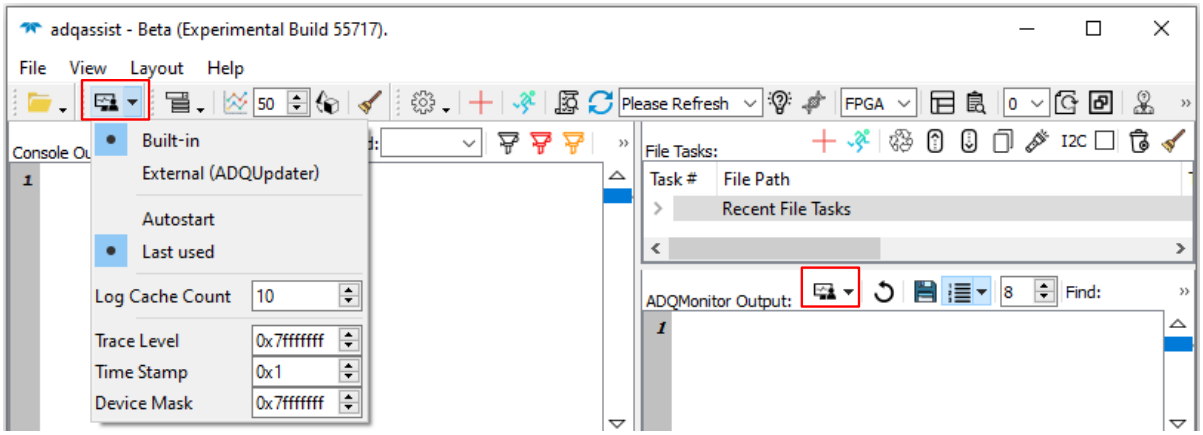

Figure 9: ADQMonitor button can be accessed via the main toolbar or the ADQMontor view. Clicking on the arrow next to the button will reveal available options.

5.2 Log files viewer

Apart from being able to capture log outputs from ADQAPI in realtime, ADQAssist can also be used to view text files. This feature is a complement to ADQMonitor to support viewing and examining offline log files that have been captured elsewhere. The advantage of using ADQAssist for viewing the logfiles instead of using other existing text editors is the automatic screening and annotations of warnings and errors in the files. This means that the user can very quickly identify the locations of these warnings and errors in the logs. Furthermore, text files can be viewed partially if needed. This means that it is possible to open a very very large log file for inspection without running into memory shortage issues.

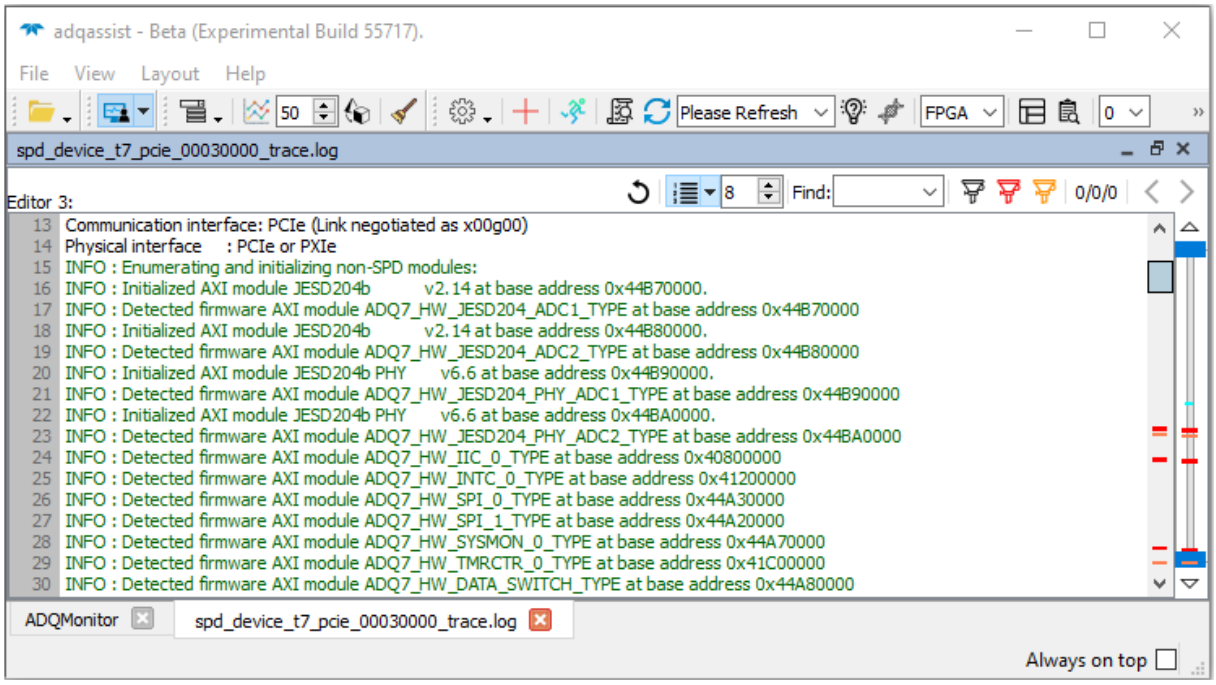

Figure 10: Warnings and errors annotations on the right-hand-side scrollbar reveal where there could be some issues.

Text or log files can be opened in ADQAssist for viewing by simply drag and drop the files into the *ADQMonitor* view or the *Console* view. They can also be opened via the *File/Open* menu or by using the popup files browser . Additional text filtering features on the editor view toolbar also helps narrowing down the interesting part of the log that might reveal possible issues.

5.3 Single device operations

Single device operations can be performed using the controls located in the larger section of the main toolbar. This toolbar and the *File Tasks* view will output all relevant information, warnings and errors into the *Console* view text area to be viewed and inspected by the user.

Note

Single device operations requires access to ADQUpdater and ADQUpdater_legacy.

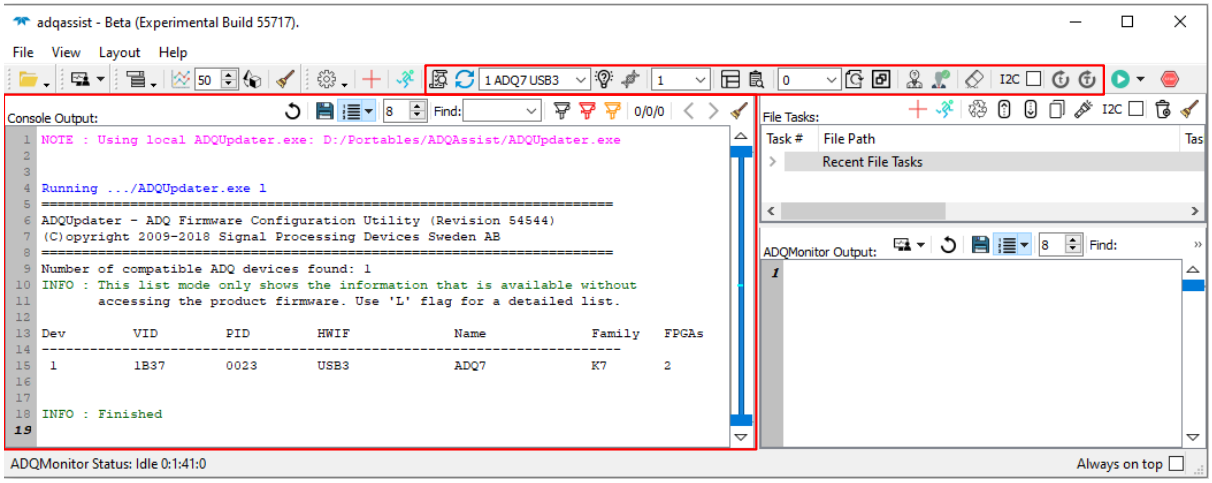

Figure 11: Simple operations that are meant to be performed on one single device at a time can be done via the highlighted toolbar.

Single device operations can be aggregated into these simple steps:

1. List available connected devices.
2. Select a device to operate on from the drop-down list. Chose Device
3. If multi FPGAs are available, select the FPGA on the device to operate on. FPGA
4. If the device supports multiple FPGA images, select the image index to operate on. Image
5. Execute desired operation by clicking on the relevant button. For example reboot or erase

Depending on the context and the selected device, some selection controls will be disabled. For example if an ADQ14 has been selected, no FPGA or image selection will be neccessary since this device family only has one FPGA and does not support multiple images on the same FPGA. Invalid selections or operations will generate errors in the *Console* view text output.

Note

User's attention to the *Console* view text output is required to determine the outcome of an operation on a device.

5.3.1 Firmware initiation test - example

This example is valid for all ADQ devices. The ADQMonitor output generated by this operation is often very useful when seeking support from TSPD. This operation can also be used to identify the physical device among all the connected devices by making the power LED of the target device flash for a few seconds.

1. Press to enable ADQMonitor to capture possible errors during firmware initiation.

2. Press to refresh the device list.
3. Select the target device from the drop-down list Chose Device.
4. Press to initiate the firmware.
5. Observe the *Console* view and the *ADQMonitor* view for any possible warnings or errors.

5.3.2 Dump device DNA - example

This example is only valid for ADQ14/ADQ7/ADQ8/ADQ12/ADQ32.

1. Press to enable ADQMonitor to capture possible errors during the request for device DNA.
2. Press to refresh the device list.
3. Select the target device from the drop-down list Chose Device.
4. Press to retrieve device DNA. It will be printed out in the *Console* view.
5. If the device is having trouble initiating the firmware to retrieve the DNA, errors will be reported in the *Console* view and the *ADQMonitor* view.

5.3.3 List FPGA images - example

This example is only valid for devices that support multiple FPGA images, such as ADQ7/ADQ8. It will also only engage ADQAPI breifly, thus ADQMonitor is optional.

1. Press on the toolbar to refresh the device list.
2. Select the target device from the drop-down list Chose Device.
3. Press to list available images. It will be printed out in the *Console* view.

5.3.4 Set default image (USB) - example

This example is only valid for devices that supports multiple FPGA images, such as ADQ7/ADQ8. It will also only engage ADQAPI breifly, thus ADQMonitor is optional. For P*Ie devices, see the next example.

1. Press to refresh the device list.
2. Select the target device from the drop-down list Chose Device.
3. Select the FPGA index 1 from the drop-down list FPGA.
4. Select the desired image index from-down list Image
5. Press to set the chosen image as the default startup image.
6. Power cycle the device to let it boot up the new default image. Alternatively press to mmediately boot into the selected image. However, this only works for USB devices.

5.3.5 Set candidate image (P*Ie) - example

This example is only valid for devices that supports multiple FPGA images, such as ADQ7/ADQ8. It will also only engage ADQAPI breifly, thus ADQMonitor is optional. This operation is similar to the USB version in previous example, except for the last few steps.

1. Press on the toolbar to refresh the device list.
2. Select the target device from the drop-down list Chose Device.
3. Select the FPGA index 1 from the drop-down list FPGA.
4. Select the desired image index from the drop-down list Image.
5. Press to set the chosen image as the candidate image.
6. Power down the PC completely and start it up again.
7. Start ADQAssist once the PC has booted up.
8. Press to list available images again.
9. Inspect the output in the *Console* view to see if the candidate image was able to load properly.
10. If the candidate image had loaded successfully, press to confirm the candidate image. This image will now be the default startup image for the P*Ie device.

5.4 Task based operations

The reason for why this section is not called *Multi devices operations* is because *Task based operations* also can perform multiple operations on the same single device as well, by scheduling the different tasks into a sequential list. For the current version of ADQAssist, *Task based operations* most of the time involves performing maintenance tasks that require an input file of some sort. For example a firmware file or a license file. But these are not the only types of operations that can be *tasked*. Advanced usage of task based operations is not in the scope of this user guide.

Note

Task based operation requires access to ADQUpdater and ADQUpdater_legacy.

Task based operations are served by the *File Tasks* view. It is a list where each row in the list represents one task to be performed. The last row in this list labeled as *Recent File Tasks* is a collapsable list of the most recent tasks that have been successfully performed. Double-clicking on this row will collapse/expand the list.

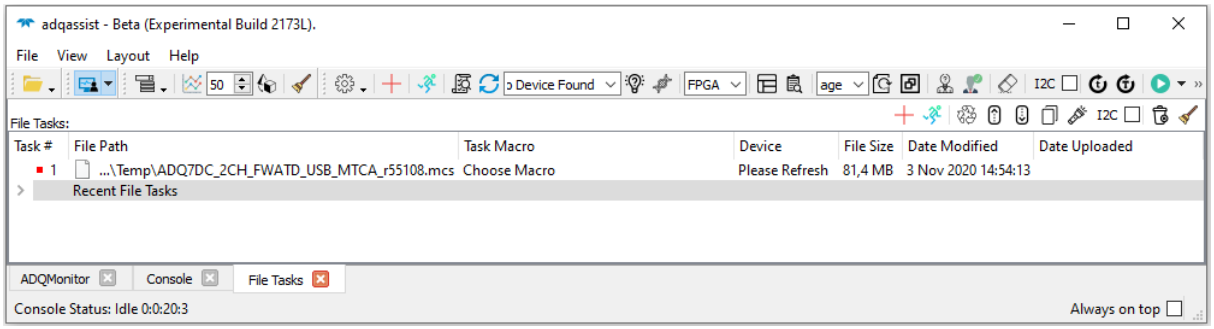

Figure 12: *File Tasks* view showing the collapsed list of most recent file tasks and a new firmware task which has not been completely configured.

A task can be created and executed thru the following steps:

1. Drag a firmware file or a license file and drop it into *File Tasks* list to create a task.
2. Click on the *Device* column of the newly created task and press to refresh the device list.
3. Select a device from the refreshed list.
4. Click on the *Task Macro* column of the newly created task and select the desired operation from the drop-down macro list. The macro text for each operation is pretty self explanatory.
5. Press to execute all scheduled tasks in the list.

5.4.1 Multiple ways to drop files into task list

Step 2 and 4 from the list above are universal and must be set for each and every new task. However, there are several ways to drop a file into the task list to create a new task for that file. The following are a few possible ways to do that:

- Drag a file from any file browser on the operation system and drop it into the list (as described above).
- Open a file via the *File/Open...* menu on the topleft corner of ADQAssist.
- Use the popup file browser from the toolbar to quickly access the file system. A file can then be dragged from this popup file browser to the tasks list. Alternatively right-click on a file and select *Add to File Tasks*.

Note

The popup file browser is a very convenient way to quickly access the file system when creating new firmware/license tasks or to access data files for plotting with ADQAssist plot feature.

5.4.2 Reusing old tasks

Tasks that have been successfully performed are added to the *Recent File Tasks* list and can be reused as is, or as a template for a completely new task. When reusing a previous task as is, the associated firmware or license file must still exist in the same location it was last used in a task. Otherwise you can modify the reused task to adapt to the new conditions. To reuse a task, do one of the following:

- Press the new task button + to reuse the latest successful task. If there are no successful tasks in the past, a completely new task will be created.
- Right-click on any task in the expanded list of *Recent File Tasks* and select *Reuse task*.

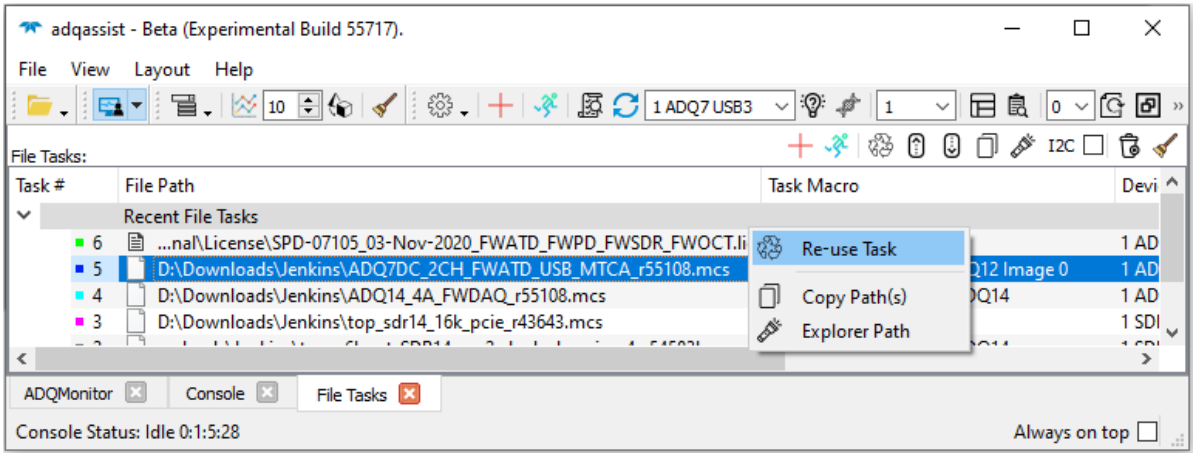

Figure 13: Press *New task button* or right-click on a previous task to reuse it.

Note

Reusing previous tasks can save a lot of time because you don't need to manually locate the same file again. Modify the reused task if needed.

5.4.3 Update firmware image 2 on ADQ7 - example

This example is also valid for devices that supports multiple FPGA images, such as ADQ7/ADQ8. It will also only engage ADQAPI breifly, thus ADQMonitor is optional.

1. Press ↻ on the toolbar to refresh the device list.
2. Select the target device from the drop-down list Chose Device.
3. Press to list available images on the device. It will be printed out in the *Console* view.
4. Inspect the output in the *Console* view and verify that the default startup image is *NOT* currently set to same image index that is going to be updated.
5. If the current default image is set to image 2, follow the example 5.3.4 or 5.3.5 to change default startup image to any other available image.

6. Use any method described 5.4.1 to add a firmware file for ADQ7 into the *File Tasks* list.
7. Click on the *Device* column of the newly created task and press to refresh the device list, if the list is empty.
8. Select an ADQ7 device from the device list for the task.
9. Click on the *Task Macro* column of the task and select the ”*Update ADQ7 image 2*” macro from the drop-down list.
10. Press to start uploading the new firmware file to image index 2.

5.4.4 Update firmware on ADQ14 - example

This example is also valid for older device family such as V6 and V5 devices that are connected via USB. It will also only engage ADQAPI breifly, thus ADQMonitor is optional.

1. Use any method described 5.4.1 to add a firmware file for ADQ14 into the *File Tasks* list.
2. Click on the *Device* column of the newly created task and press to refresh the device list, if the list is empty.
3. Select an ADQ14 device from the device list for the task.
4. Click on the *Task Macro* column of the task and select the ”*Erase & Update ADQ14*” macro from the drop-down list.
5. Press to start uploading the new firmware file.

5.4.5 Upload license file to ADQ7 - example

This example is valid for ADQ14/ADQ12/ADQ7/ADQ8. ADQMonitor should be turned on for this operation to detect any possible issues. License files are required for specialized firmware such as FWATD or FWPD. They are not free and must be purchased via TSPD. License files are tied to a device DNA, thus make sure to provide the DNA and the serial number of your device when requesting a license file for a specialized firmware. See 5.3.2 on how to obtain a device DNA.

1. Use any method described 5.4.1 to add a license file for ADQ7 into the *File Tasks* list.
2. Click on the *Device* column of the newly created task and press to refresh the device list, if the list is empty.
3. Select an ADQ7 device from the device list for the task.
4. Click on the *Task Macro* column of the task and select the ”*Upload License*” macro from the drop-down list.
5. Press to start uploading the license file to the device.

6 Basic data visualization

ADQAssist comes with an advanced data visualization feature often used by TSPD staff internally for testing or on-site debugging situations. This feature can also be very handy for new ADQ users to quickly get a visual look at the data that they have acquired with their own code. For example when testing out one of the many C-code examples to collect some data from an ADQ device. The data visualization feature can plot any binary data file in any sample format. Ascii formated data files can also be plotted but the data samples must be arranged in a single column, and each sample must be separated by a newline. Hence it is recommended to save any acquired data in binary format whenever possible since it would make things much easier for any data visualization tool.

The data visualization feature is served by 3 views. The *Plot Item List* view, the *Signal Plot* view and the *DSP Plot* view. To see all these views at the same time, go to menu and select *Layout/Restore Layout/Data Plot*. The *DSP Plot* view is still in its experimental stage and will not be covered in this document.

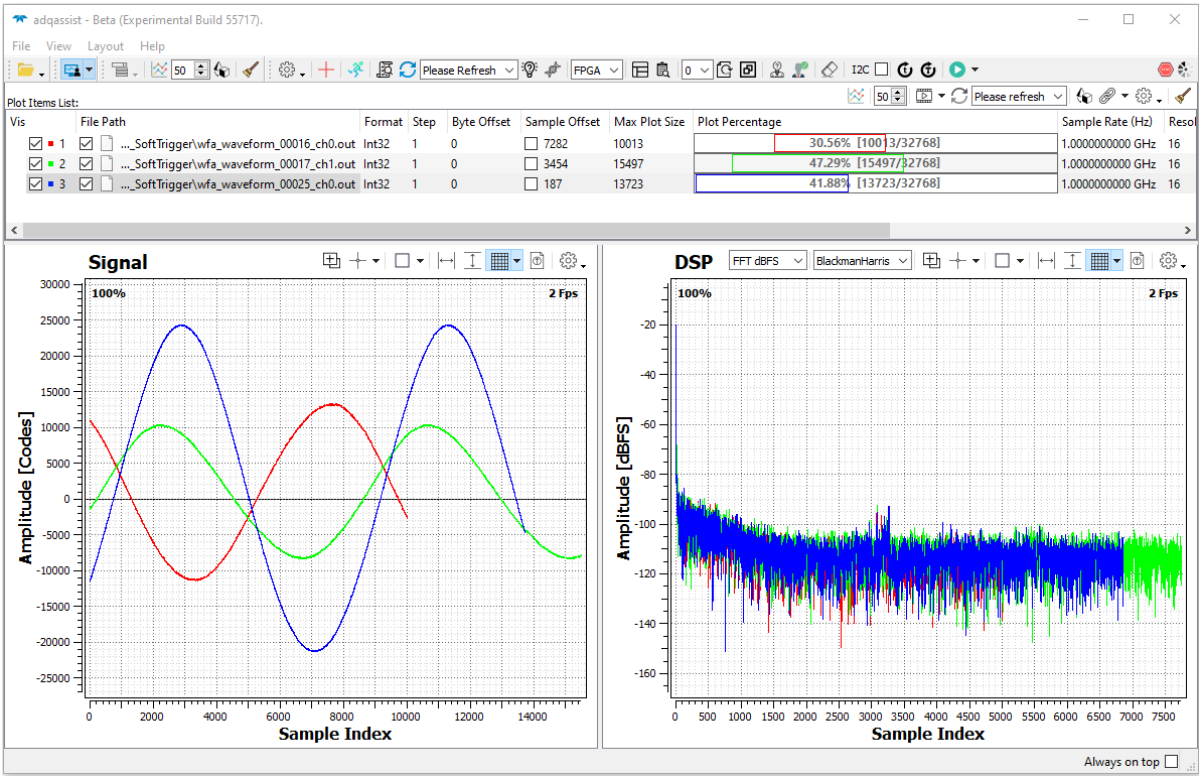

Figure 14: Data visualization is served by these 3 views. They can be shown at the same time via the *Data Plot* layout preset from the *Layout* menu.

6.1 How to plot

Plotting a file can be as simple as dragging a file and dropping it into one of the three mentioned views. A plot item will be created for that file in the *Plot Items* view. The popup file browser described in

5.4.1 can also be used here to quickly access available data files on the file system for plotting. Either drag and drop files from this popup file browser into one of views, or right-click on the files of interest to see additional plotting options.

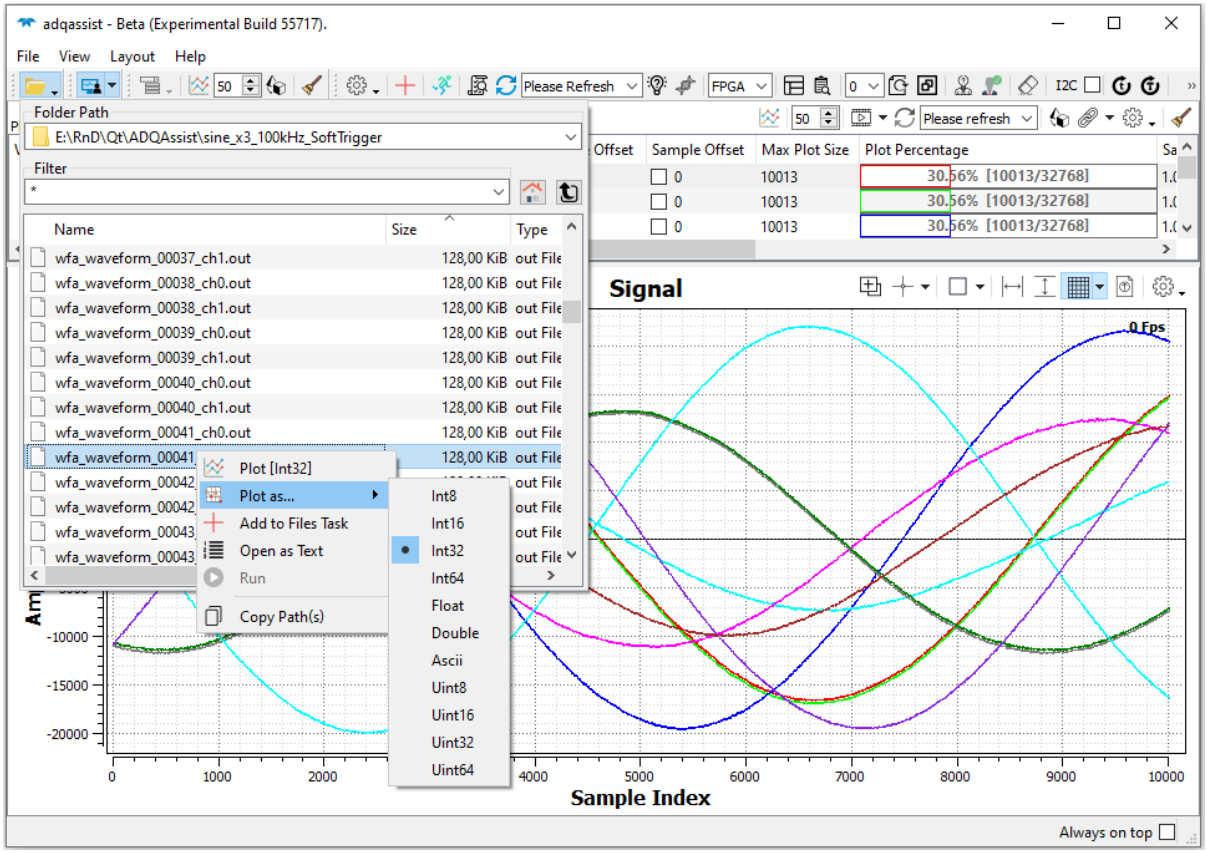

Figure 15: The popup file system browser can also be used to quickly access data files for plotting.

ADQAssist can change the plot sample format for binary data files on-the-fly and it is also possible to selectively plot only a part of a larger file to avoid overloading the system when opening a very large data file. All files are plotted on the same plot to make it easier to compare different data sets, but the user can chose to show or hide the plot for each file. If different data sets need to be visualized on separate different plots, multiple instances of ADQAssist can be used to accomplish this.

6.2 Default plot item values

Every file that is added to the *Plot Items* view is visually plotted with a set of default parameters. These default parameters can be changed via the popup plot item settings panel ⚙ on the top-right corner of the *Plot Items* view. Once a file has been plotted with these default settings, the indiviual parameter for that plot item can then be changed by clicking on one of columns for that plot item.

Note

Every new plot item is initially plotted with a set of default parameters. These default parameters can be changed via the plot item setting ⚙ panel.

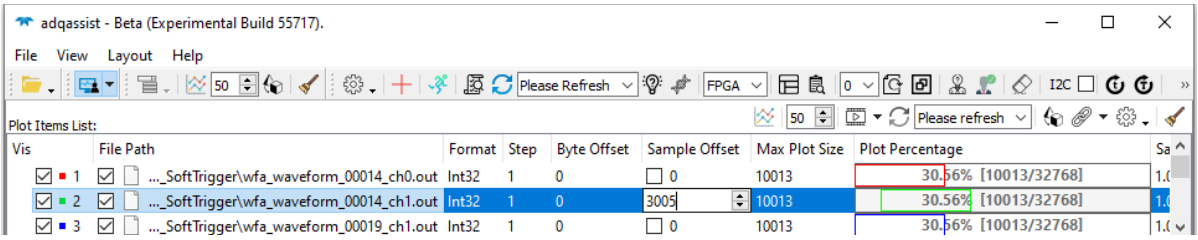

Figure 16: Inidivual plot item's parameter can be changed by clicking on the corresponding column and modify the values.

6.3 Plot item columns

Each plot item in the *Plot Items* view has a set of columns which represent the parameters for the item. Most of these columns are editable and changing the values in these columns might modify the plot. However, some of the columns are locked and some will not have any impact on the plot itself due to the analog nature of these parameters. For example changing the sample rate of a plot item will not affect how the plot looks since it would only affect the time axis value of that particular plot. And this is one of the more advanced usage which will not be covered here.

For basic plotting usage, the most relevant columns of a plot item that the user can interact with and modify are following:

- Format
- Step
- Byte Offset
- Sample Offset
- Max Plot Size
- Plot Percentage

The *Plot Percentage* column is particularly useful since it is an interactive spanslider which can be used to quickly navigate which part of a file the user wants to visualize. Dragging this slider or adjusting its width will change which part of the file will be visualized into a plot. Alternatively modifying the *Sample Offset* and/or *Max Plot Size* column will give the same effect. If the plot does not look as expected, verify if the *Format* column has the correct type.

Note

Check if the *Format* column has the correct type if the plot looks not as expected.

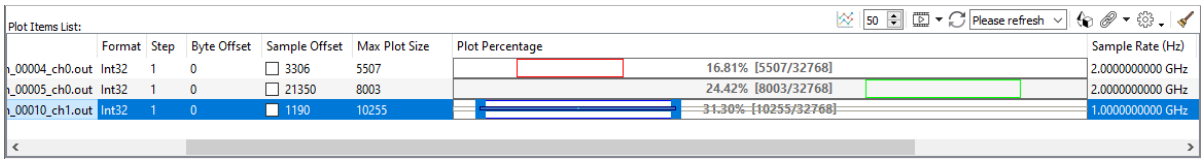

Figure 17: The interactive slider in the *Plot Percentage* column can be adjusted to selectively plot part of a file.

6.4 Plot interactions

6.4.1 Plot zoom

The plot can be zoomed in by drawing a rectangle on the plot canvas with the left mouse button pressed. This is called *boxed zoom* and can be done repeatedly by drawing another rectangle inside the already zoomed area. Boxed zooms are stacked and can be navigated backwards by pressing the right mouse button anywhere on the plot canvas. Each press will go back one level of the stacked zooms untill no zoom level is left in the stack. Stacked zooms cannot be navigated forward, only backward. Thus a new selection retangle must be drawn to zoom in again.

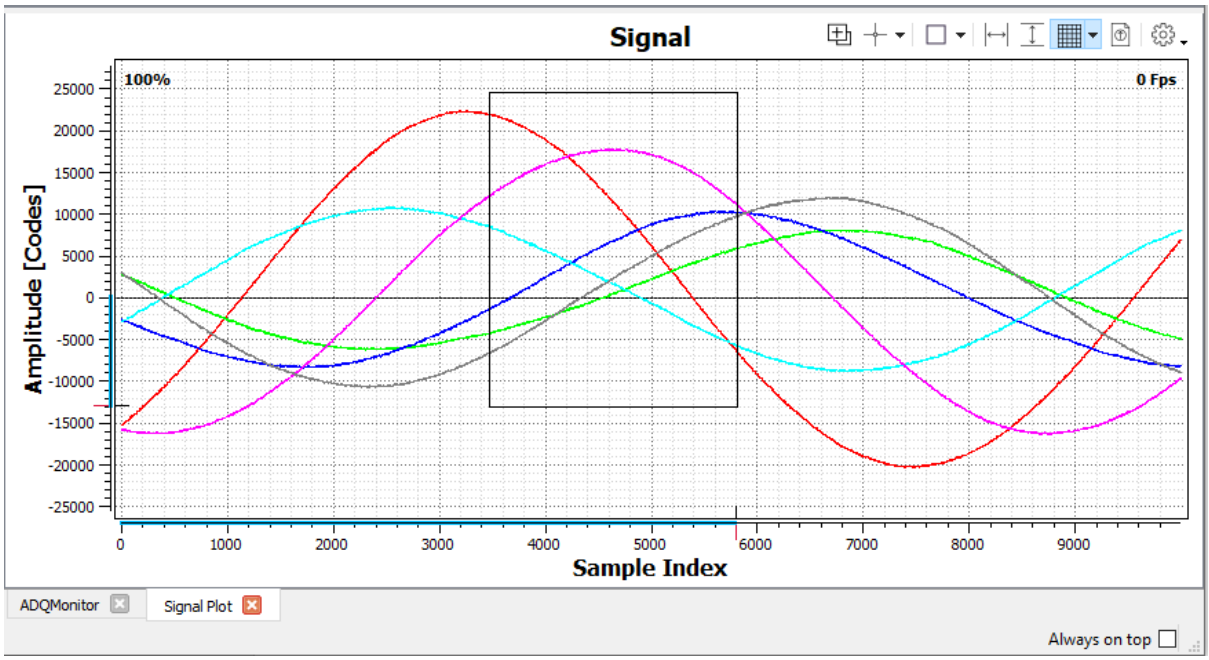

Figure 18: Draw a rectangle with the left mouse button to zoom into an area of interest.

Zoom can also be performed via the mouse scroll. The zoom will be centered around the mouse pointer location on the canvas. Scroll zooms are not stacked, but the right mouse button can still be used to go back to the original un-zoomed view of the plot. Scroll zoom is recommended only to fine-tune the zoom level after using boxed zoom.

6.4.2 Curve highlight and selection

A curve can be highlighted by hovering the mouse pointer over any visible curve on the plot canvas. Apart from the highlight, the closest sample point on the curve will also be annotated. To select that curve, press the left mouse button while the curve is being highlighted. The curve is now persistently highlighted and a tracking point will follow the mouse horizontally along the curve. While a curve is selected, clicking anywhere in the canvas will drop a marker point on the curve corresponding to the x-coordinate of the mouse pointer. To deselect the curve and turn off the persistent highlighting and tracking, press the left mouse button anywhere outside the plot canvas, but within the plot view. For example, below the x-axis.

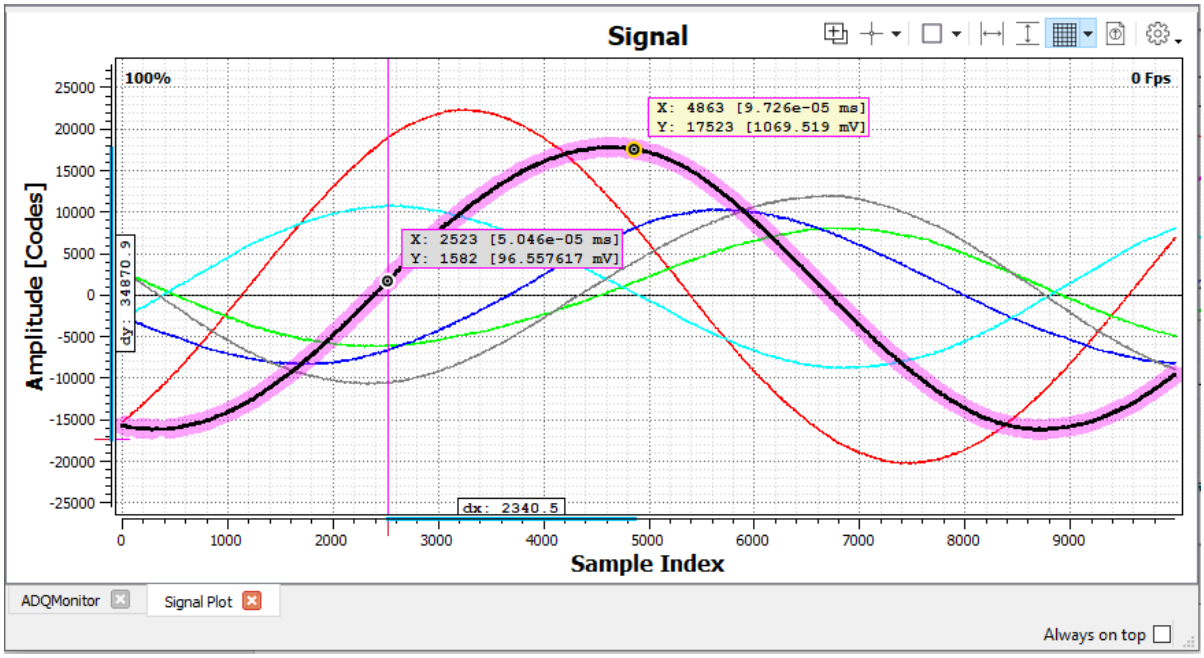

Figure 19: A curve has been selected, with a stationary marker point and a moving tracker point.

A Appendix

Table 3: Supported Linux Distributions

Distribution Name	Version	Platform
Ubuntu	16.04	x64
Ubuntu	18.04	x64
Ubuntu	19.04	x64
Ubuntu	20.04	x64
Debian	9.0	x64
Debian	9.0	x64
CentOS	7	x64
CentOS	8	x64
CentOS	8 Stream	x64
ScientificLinux	7	x64
RHEL	7	x64
Fedora	29	x64
Fedora	30	x64
Fedora	31	x64
Fedora	32	x64
SLE	15	x64
SLE	15 SP1	x64
openSUSE Leap	42.1	x64
openSUSE Leap	42.2	x64
openSUSE Leap	42.3	x64
openSUSE Leap	15.0	x64
openSUSE Leap	15.1	x64
openSUSE Leap	15.2	x64

TELEDYNE SP DEVICES
Everywhereyoulook™

Worldwide Sales and Technical Support
spdevices.com

Teledyne SP Devices Corporate Headquarters
Teknikringen 8D
SE-583 30 Linköping
Sweden
Phone: +46 (0)13 645 0600
Fax: +46 (0)13 991 3044
Email: spd_info@teledyne.com