DPScope User Manual

Version 1.0a
January 30, 2010

1 Introduction

The DPScope is a low-cost, microcontroller based digital oscilloscope intended for hobby and educational use. If you are new to oscilloscopes, we recommend the following introductory application notes (created by Tektronix, one of the major oscilloscope manufacturers for the professional, high-end market) which will supply all the knowledge necessary to understand this user manual and get the most from your DPScope:

The XYZs of Oscilloscopes
The ABCs of Probes

You can download both documents either from the Tektronix website (http://www.tektronix.com) or from the DPScope website (http://www.dpscope.com → Downloads).

If after reading through the documentation you still have questions regarding the DPScope or want to provide feedback or suggestions for improvement, do not hesitate to contact us at

support@dpscope.com

2 Software Startup

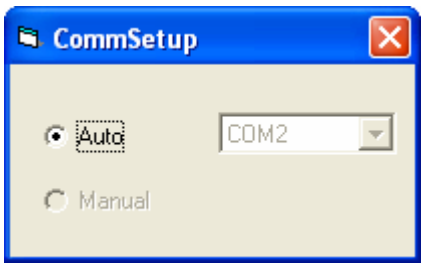
After you installed the USB driver and the DPScope software (see the assembly guide for details) plug in you DPScope and wait a short time so the computer has a chance to recognize the scope. The DPScope's power LED on the front panel should blink quickly a few times and then stay solidly on – this is the sign that the scope has successfully powered up

Then launch the PC software (Start → DPScope → DPScope). After a few seconds the DPScope program should appear on the screen. If the program cannot connect to the DPScope you will get a popup (see below) that you need to confirm in order to continue. The software will still start after that, but you will not be able to run the acquisition. In this case, close the software, unplug the oscilloscope, plug it in again (you may want to try a different USB port if that does not help), and launch the software again.



Unless you added the external supply option (contact us for details on this) the DPScope should always be attached to a USB port directly at the computer, or through a powered USB hub. Unpowered USB hubs tend to have excessive supply voltage drop and this would directly impact the DPScope's vertical accuracy (the DPScope relies on the assumption that the supply voltage is exactly 5V).

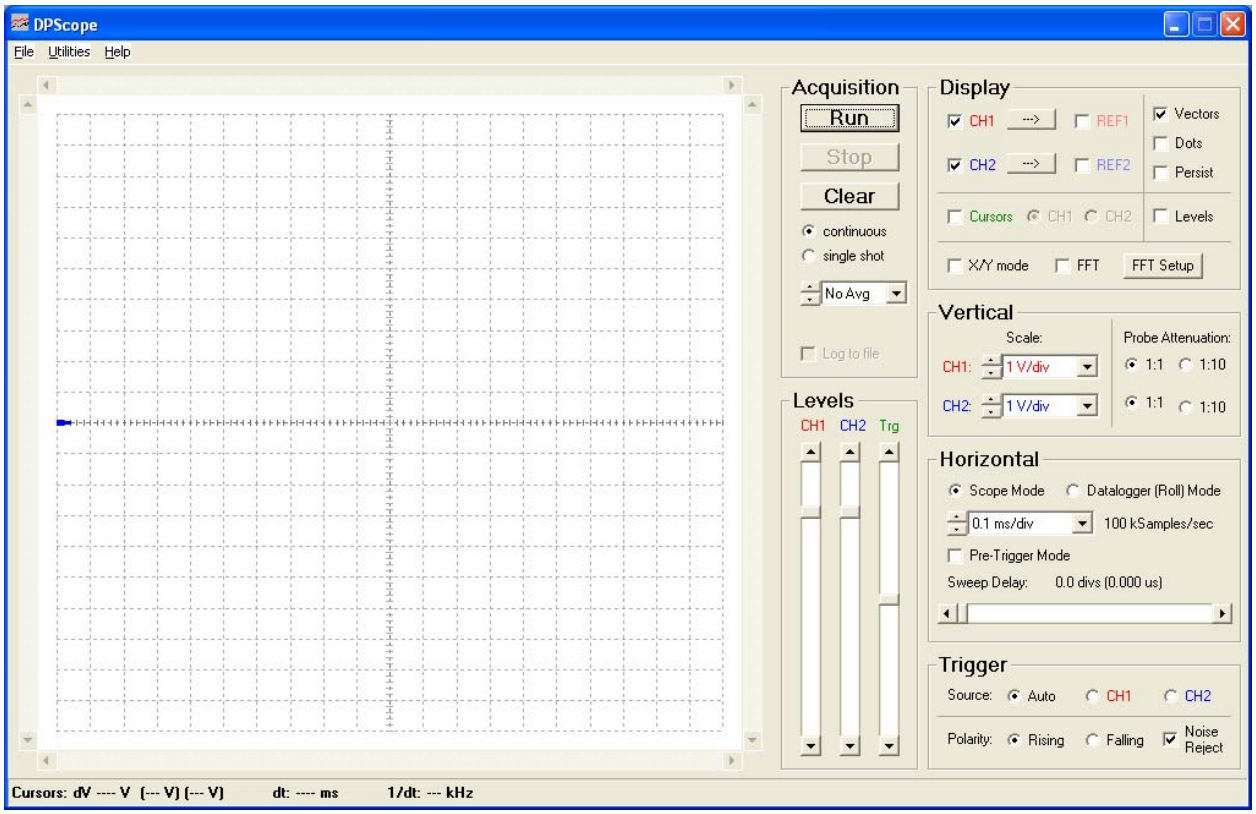
The scope connects through a so-called "virtual COM port" (VCP), meaning that while the physical connection is USB, the interface driver makes the connection look like a legacy RS-232 COM port. You can see such ports under "COM and LPT" ports in the Windows device manager. To find out which virtual COM port the DPScope uses, go to Utilities → Comm Setup in the DPScope program (the software will automatically detect which port the DPScope is connected to; in the example below it is COM2, but it can be any port between COM1 and COM99):



The **minimum required screen resolution** for the DPScope software is **800x600 pixels**. The screen layout will adapt itself – resulting in a larger display with more widely spaced and larger controls) in case a bigger screen (1024x768 pixels or larger) is available; apart from that the available controls, layout and functionality is the same in either case.

3 The Main Screen

After startup the screen should look as shown below:



The left side of the display is taken up by the waveform display. On the right side you can see that the controls are grouped in several functional blocks (we will explain each of them in more detail later):

- **Acquisition:** Main control of acquisition process
- **Display:** Determines how the acquired data is to be displayed.
- **Vertical:** Controls the input amplification/attenuation for each of the two input channels.
- **Levels:** Sets the voltage offset for each channel, and the trigger threshold level.
- **Horizontal:** Controls the sampling speed, delay, and sampling method.
- **Trigger:** Defines the trigger source.

On the bottom there is a status line that displays the cursor information (right now the cursors are turned off), which you use to make measurements on the displayed waveforms (e.g. to determine the signal amplitude or frequency).

On the top of the screen you find the menu containing three entries, File (to save the displayed waveforms in numerical format, and to save and restore the DPScope's software settings), Utilities (interface setup, and probe calibration), and Help (displays general info about the software).

4 First Acquisition

After startup the software is set up in a way to maximize you chance to see *something* useful on the screen right away: The voltage range is set to maximum, ground level centered on the screen, continuous mode and trigger in Auto mode (meaning the acquisition will run without waiting for a trigger event), and so on.

Attach the probes to some signal source, e.g. the toggling output of a microcontroller. Press the “Run” button. You should see a waveform on the screen. Since the trigger mode is “Auto”, most likely the waveform will not be stable but rather jump around.

Now play around a bit with some of the controls. Don’t be afraid! You can’t break the scope no matter what settings you change. The worst that can happen is that you no longer acquire a waveform and/or see it on the screen. If you get completely stuck, simply close down the DPScope program and start it again.

Move the offset sliders and see how the waveforms follow. Now move the trigger level slider so the trigger level (indicated by a green triangle on the left border of the waveform display) is somewhere in the middle of the CH1 waveform (the red waveform). Now select “CH1” in the trigger control menu. That waveform should now be stable (assuming you waveform is periodic). Try switching between rising and falling edge trigger.

Now change the sample rate in the Horizontal control display and see how the waveform changes (it’ like zooming in and out on the waveform timing). Move the Sweep Delay slider and the waveform moving left and right.

In the Display control box, turn the waveforms on and of by clicking on the appropriate checkbox (you can still trigger on a waveform even when it is not displayed). Change the display style to points and/or infinite persistence.

Congratulations – you have just measured your first signal with the DPScope!

5 General Remarks

The DPScope is a digital storage oscilloscope (refer to the “XYZs of Oscilloscope” mentioned in the introduction if you are unsure what that is), based on a Microchip dsPIC 16-bit microcontroller. Most of the oscilloscope hardware is actually integrated in this single chip – analog-to-digital converters, sample logic, trigger logic and threshold generation, memory, PC interface. Only the analog frontend (input amplifier/attenuator, vertical offset generation) consists of external components.

5.1 Acquisition Modes

The DPScope’s acquisition engine has four different modes (again, refer to the “XYZ of Oscilloscopes” document for further details):

5.1.1 Normal Mode:

Acquisition is started by a trigger event (or automatically right after the previous acquisition if the trigger mode is “Auto”) and samples the signal in real time. This is very similar to how a classic analog oscilloscope operates. Real-time acquisition means the acquisition is very fast (up to approx. 40 records per second on a sufficiently fast computer) and you can capture single-shot events. (Note that the acquisition rate will decrease for slow sample rates – slower than 10 kSamples/sec or 1 ms/div – because it takes some time to simply capture each full record if only a few samples are taken per second). Also, since this mode uses an interrupt generated by the internal hardware comparator the timing is very tight (i.e. reaction to the trigger event is almost instantaneous), resulting in very low jitter against the trigger – in practice that means there is very little horizontal (timing) “wiggle” of the displayed waveform.

This mode has two limitations: First the maximum sample rate is 1 MS/sec (1 million samples per second), because the controller has to digitize and store the data as fast as it comes in (up to two Megabytes of data per second since there are two channels!). Second, since the trigger *initiates* the capture process, you can only look at things that happen *after* the trigger unless you can supply a suitable pre-trigger signal to the second channel to trigger on. If the waveform is periodic you may be able to use the Sweep Delay feature to trigger on one edge but look at the following one to get around this limitation.

5.1.2 Pretrigger Mode:

In this mode the acquisition runs continuously until the trigger event (e.g. the waveform crossing the trigger threshold from below, i.e. a rising edge). After the event it keeps sampling for a user-defined time (between zero and one full record length) and then stops, transferring the captured data to the computer. This means you can now also look at things that happen *before* the trigger, which often is the interesting part (e.g. the things that precede a wrong data bit). Since the acquisition is again happening in real time in this mode you can look at single shot (non-repetitive) signals, and data acquisition is very fast again. Sweep Delay is not available in this mode since this feature would not make much sense (after all, if you want to look at things happening a while after the trigger it is better to use Normal Mode anyway).

The additional versatility of this mode comes with two tradeoffs compared to normal mode. First, the sample rate is limited to a maximum of 200 kSamples/sec (because in addition to sampling and storing the signal the scope now must continuously look out for trigger events at the same time). This is still good enough for the audio range (up to about 20 kHz). Second, since the acquisition is completely asynchronous to the signal to be measured, there is a ±1 sample uncertainty as to how the actual trigger lines up to the sample timing. This results in increased trigger jitter, i.e. the waveform on the screen will wiggle back and forth by ± one sample interval (±1/10th of a division).

5.1.3 Equivalent Time Sampling Mode:

This mode works similar to Normal Mode in the sense that the acquisition is initiated by the trigger event, so again you can only look at things that happen some time after the trigger. But other than Normal Mode in Equivalent Time Sampling Mode the scope makes no attempt to

sample the signals in a single sweep. Instead it uses its maximum real-time sample rate (1 MSample/sec) to capture a sample every microsecond, then waits for the next trigger and captures a second record, and so on. For each record it waits for a slightly longer time after the trigger before it starts sampling; this delay can be controlled in increments much finer (down to 50ns or 1/20th of a microsecond) than the smallest real-time sample interval (1 μs). That way – assuming the waveform is repetitive with respect to the trigger – the sample points of subsequent record will interleave each other with respect to the actual signal timing. Plotting them together (suitable shifted by the respective added delay versus the trigger) produces a picture of the signal with finer resolution than possible with real-time sampling. Once the delay has spanned one full real-time sampling interval (one microsecond), the DPScope restarts the acquisition delay from zero delay.

The main advantage is that you can resolve the signal in much finer increments (up to 20 MSamples/sec, or equivalent-time sample intervals of just 50 nanoseconds) than possible with real-time sampling.

There are a number of tradeoffs connected with equivalent time sampling. First there is lower acquisition speed (because the DPScope had to acquire and transfer two or more data records for each new waveform on the screen), and the necessity that the signals be repetitive with respect to the trigger. Second you always need to use a trigger to get usable waveforms, otherwise subsequent partial captures will be unrelated timing-wise, and the composite waveform on the screen will look like random noise rather than the waveform you expect. Also it's obvious that you can't make single-shot acquisitions in this mode (the software allows single short mode, but here it means that one full waveform – consisting of several partial captures – gets acquired and then the acquisition stops). Still, given the high acquisition speed (40 frames/sec) of the DPScope and the fact that most signals one encounters are repetitive (or can be made repetitive) anyway means these restrictions are far less serious than you might think.

5.1.4 Datalogger (Roll) Mode:

For very slowly varying signals it takes a long time before a full record has been acquired. E.g. at 1 sec/div timebase each acquisition (20 horizontal divisions) takes 20 seconds, plus the time it takes before the first trigger event after the end of the previous acquisition happens, so you may wait close to a second before the screen refreshes just once. In this case it is usually better to display the sampled data as it comes in. This is what the Roll Mode is for: The scope acquires the signals continuously and plots the samples on the screen right away. The waveform scrolls to the left and the new points get added to the right, similar to how a classic datalogger would draw curves on a moving band of paper from a roll (that's where the Roll Mode has its name from).

There is no trigger available in this mode (since we do not want to wait for a trigger event anyway). But you can log the captured data into a text file on your hard drive, so unlike the other three modes you can capture data sets of arbitrary length, only limited by the free space on your hard drive. Clearly this mode only makes sense for relatively slowly varying signals, so the sample rate is limited to 20 samples per second (timebase 0.5 sec/div) or slower.

5.2 Probe, Vertical Gain Setup and Trigger Setup:

When you want to acquire a new (unknown signal), it is best to maximize you chances of seeing *something* before fine-tuning your settings. Choose the widest voltage range available (1 V/div) and set the channel offset to the center of the screen. That way your visible range spans from -10V to +10V which will capture most signals you encounter. For larger signal, use a 1:10 probe to widen the range even further.

Whenever you use a new 1:10 probe, first perform a probe calibration (described in the DPScope Assembly Guide). 1:1 probes should not need any calibration because the DPScope's input has been calibrated for them as part of the scope assembly.

It is usually best to set the trigger to "Auto" mode – otherwise you'd have to hunt for a valid trigger level on the invisible waveform before the scope displays anything. The waveform timing will not be stable that way (but will move randomly in time), but you will be able to see which voltage range the signal spans. Using the "Persistence" display mode can sometimes be helpful if looking at rare spikes where the signal amplitude is difficult to judge otherwise.

Once you know the range you can adjust vertical gain and offset to center the signal vertically and make it fill a good portion (at least a quarter) of the vertical display range.

To set up the trigger, move it to the center of the trigger channel's signal band (i.e. the green triangle on the left should be somewhere between maximum and minimum level of the signal – CH1 or CH2 – you want to trigger on) and turn the trigger on. Switching on the "Levels" display can be helpful there as well.

Now you can fine-tune your setup (e.g. use a different sampling mode, change the trigger polarity, etc.). When you expect to measure the same signal again later you may want to save the complete setup to a file (File → Save Setup) that can be loaded back later. That way you do not have to repeat the full setup procedure over and over.

5.3 Undersampling / Aliasing

One very common trap that many beginners (but not only they!) fall into with digital sampling oscilloscopes is so-called aliasing. This effect can happen whenever the sample rate is too slow compared to the signal frequency. To pick a simple example, assume you have a 1 kHz sine wave that you sample with a 1kSample/sec sample rate. In this case each acquisition will hit the same point in the signal period (just one period later each time), so the displayed waveform on the screen will be a flat, horizontal line rather than a sine wave – clearly wrong! If the signal frequency is a steady 1.1 kHz and the sample rate is 1 kHz each sample will be $1/10^{th}$ of a period offset on the, and the resulting displayed waveform will be a very convincingly-looking 100 Hz sine wave – this can be very misleading! (Incidentally, this is in fact a variation of the equivalent-time sampling process, and is called "coherent undersampling", and is widely used whenever one can't get sufficient sample rate to sample incoming high-speed signals in real time).

The solution is of course to sample the waveform with a sufficiently high sample rate. So if in doubt, change the sample rate and see if the waveform behaves "normally" (e.g. when doubling the sample rate the displayed frequency of the signal should not change). And as a general

guideline when working with an unknown signal always start looking at it with maximum sample rate and only then change it to something slower.

6 Description of User Controls

6.1 Acquisition



6.1.1 Run

Starts the acquisition process – data is captured (if trigger event occurs) and displayed on the screen. If “continuous” is selected the acquisition will repeat until “Stop” is pressed. In case of “single shot” acquisition the scope acquires and displays only one record and then stops if “No Avg” (no averaging) is selected; otherwise it will acquire the given number of records to average over and then stop. This means that if you want to do a true single shot capture you need to set the averaging to “No Avg”.

In datalogger (roll) mode single shot and averaging is not available; in this case the Run button simply starts the continuous acquisition.

6.1.2 Stop

Stops the acquisition process.

6.1.3 Continuous / Single Shot

Selects whether the DPScope shall continuously acquire new waveforms, or only acquire once and then stop. Continuous acquisition is useful for repetitive waveforms when you want to see if

or how the signal changes over time. Single shot capture is useful e.g. for event that happen only once (e.g. a sudden pulse), or when you want a steady picture to take measurements on.

6.1.4 Averaging

Selects how many waveforms to average over (1, 2, 5, 10, 20, 50, or 100). Averaging reduces random noise on the waveform, resulting in a much cleaner picture that allows you to see small details that may otherwise be hidden under the noise. For averaging to make sense you need to have a basically stable, repetitive waveform, so you will need to turn the trigger on (i.e. auto mode won't work because the sample timing would be asynchronous to the signal period).

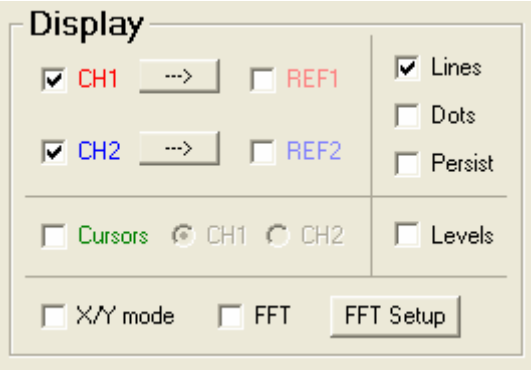
Note that the averaging process used is not a simple arithmetic average over the given number of samples; instead it uses exponential weighing, i.e. earlier acquisitions have less weight than the more recent ones. This is the digital equivalent to an analog R-C low-pass filter (but unlike a filter in your signal path it will *not* degrade your signal bandwidth or rise time; it will just filter out fast changes from one capture to the next, but not fast changes along the waveform).

Averaging is only available in scope mode, not in datalogger mode.

6.1.5 Log to File

This function is only available in datalogger mode. It allows you to write all the captured samples directly into a file. This way you can record signals of arbitrary time spans and with an arbitrary number of sample points, no longer restricted to the DPScope's 200 points per channel record length.

6.2 Display



6.2.1 CH1 / CH2

Turns the display of scope channel 1 and 2 on or off, respectively. Note that this does not affect the signal capture itself, meaning you can still trigger on a channel even when it isn't displayed.

6.2.2 REF1 / REF2

Turns the display of reference waveform 1 and 2 on or off, respectively. To copy the currently displayed waveform (CH1 or CH2, respectively) into the respective reference waveform (REF1 or REF2, respectively) you use the appropriate "→" buttons.

Reference waveforms are very helpful e.g. to judge small signal changes by comparing the stored waveform to the displayed waveform.

Another important application is when you want to compare more than two signals; one example would be an SPI bus consisting of clock, data, and chip select. In this case you can first set up the DPScope so CH1 looks at the chip select and CH2 looks at the clock signal, and it triggers on the falling edge of the chip select signal. Capture both signals with single shot mode, and save CH2 to REF2. Now connect CH2 to the data line instead, and repeat the capture. You now have all three signals (clock, data, chip select) on the screen at the same time.

6.2.3 Vectors

Checking this option will cause the scope to connect the captured data points with lines. This makes the actual waveform much easier to see, especially when there are fast changes (which would make subsequent points fall on places far away on the screen).

6.2.4 Dots

Checking this option will cause the scope to indicate the actual sample points with small circles. This can be very useful if you want to check if your sample rate is truly high enough for an accurate waveform display. Second, use this mode (and turn "Vectors" off) when you are in X-Y display mode – if the signals change fast compared to the sample rate, connecting subsequent sample points with lines can result in a messy, difficult-to-interpret picture on the screen.

6.2.5 Persist

Normally the screen is cleared whenever the scope has a new acquisition is to display. Checking this option will make the scope keep all waveforms on the screen until you either press "Stop" and then "Start " again, or until you press "Clear". This allows you to see over which range the waveform is changing; important cases are measurement of peak-to-peak noise or timing jitter. This is also a great tool to capture rare glitches (sudden waveform aberrations) that you may miss otherwise because without persist mode they may disappear from the screen again so quickly that you can't see them. When persist mode is turned on however even a single such event with remain easily visible.

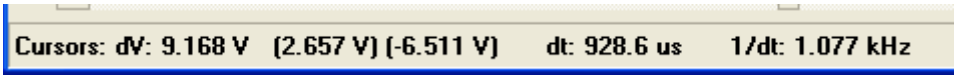
6.2.6 Levels

Turns level markers (and the trigger position marker in pretrigger mode) on and off. While the levels (CH1 and CH2 offsets, and the trigger threshold) are also indicated with colored triangle markers on the left edge of the waveform display, sometimes it is helpful to have rulers across the whole screen.

6.2.7 Cursors, CH1, CH2

Turns the measurement cursors on and off, respectively, and selects which channel (CH1 or CH2) they refer to.

The cursors are the solid horizontal and vertical lines in the waveform display which appear whenever you check the “Cursors” checkbox. You can move them with the slider bars on top/bottom and left/right of the waveform display. There current level and time positions as well as the difference in their positions are shown in the status line on the bottom of the scope window:



Because the voltage scales can be different between CH1 and CH2 you need to select which of these two waveforms the cursors shall refer to. (The time scale is the same for both so this selection has no bearing on timing measurements, i.e. on the vertical cursors).

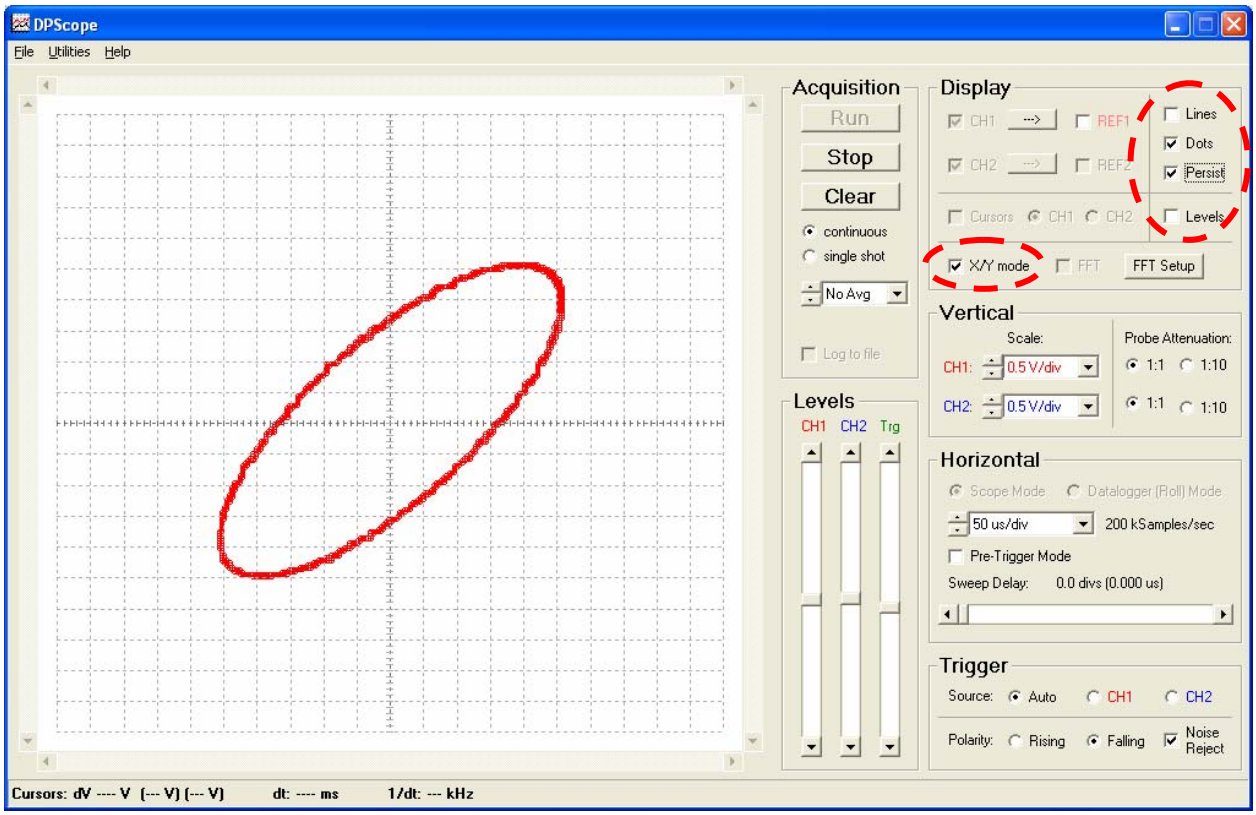
With the cursors you can perform many different measurements on the displayed waveforms, for example:

Levels: amplitude, high and low level, average level, overshoot and undershoot
Timing: period, frequency, delay, phase, rise and fall time

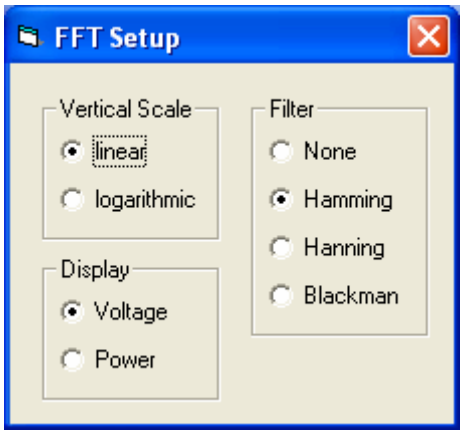
6.2.8 X/Y Mode

This checkbox changes between normal (signal versus time, or Y-T) waveform display and X-Y display mode. In X/Y mode, instead of plotting the CH1 and CH2 waveforms against time, now CH1 determines the horizontal (X) position and CH2 the vertical (Y) position of each point. A common application for this is to measure phase shifts, e.g. in order to analyze reactive elements (inductors, capacitors).

It is usually easiest to first set up the waveforms in normal (Y-T) mode and only then change to X/Y mode. In addition, usually you will obtain the best picture if you choose “auto” trigger, turn off “Vectors” display and turn on “Dots” and “Persist”.



6.2.9 FFT / FFT Setup



Checking the “FFT” option will change the display to frequency domain: The oscilloscope performs a real-time FFT (Fast Fourier Transform) on each captured record. This is a highly useful mode to analyze periodic signals or to find small periodic components (e.g. periodic noise from the power line) hidden in the main signal – it is much easier to see an isolated 60 Hz component (spectral line) in a frequency plot than it is to see it among all the random noise riding on the signal.

Note that in FFT mode the scope alternates the signal capture between channels, i.e. only captures one channel at a time. It does that so it can acquire a longer data record (410 points),

which results in finer frequency resolution. You can still freely choose which channel to trigger on, i.e. also trigger on a channel when it is not displayed).

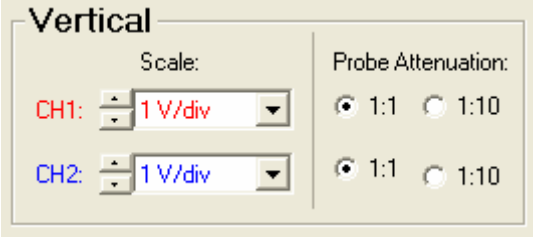
Similar to X/Y mode it is usually easiest to first set up the waveforms in normal (time domain) mode and only then change to FFT mode. Also, in most cases FFT mode works fine in “auto” trigger mode (as long as the waveform is periodic), and with sample rates low enough so a few signal periods are displayed on the screen. (However, make sure that your sample rate is at least 3-4 times your signal period, otherwise aliasing will occur, resulting in erroneous frequency domain pictures).

You can use the cursors to make measurements of frequency and relative power. Note that the sample rate box changes to frequency units (Hz and kHz).

Pressing the “FFT Setup” button brings up a small panel where you can modify the settings used for the FFT conversion: Linear (V) vs. logarithmic (dB) scaling, voltage vs. power, and the filter type to use. (The filters reduce artifacts caused by the finite length of the data set. The available types from top to bottom – none, Hamming, Hanning, and Blackman – result in increasingly accurate amplitudes for the price of increased spectral line width).

At the moment FFT mode is only available in real-time sampling mode, i.e. for sample rates up to 1 MSample/sec without pretrigger.

6.3 Vertical



6.3.1 Scale

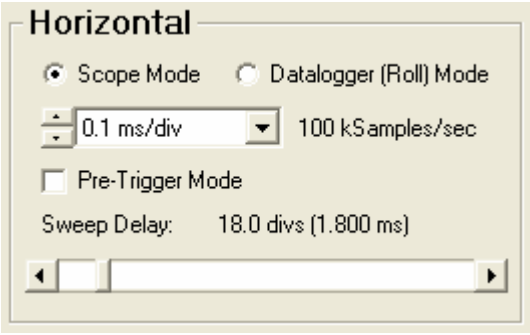
The vertical gain boxes set the input amplifier gain, so you can look at very small (Millivolts) as well as quite large (several Volts) signals. The scaling is displayed in V/div (Volts per division) or mV/div (Millivolts per division), so e.g. at 1V/div each vertical unit (interval between two horizontal dashed grid lines) corresponds to 1 Volt. Except for the 5mV/div range (50mV/div with 1:10 probe) the settings for CH1 and CH2 are independent from each other. The 5mV.div range however uses the high-resolution mode of the DPScope’s digitizer and is only available for both channels at the same time.

6.3.2 Probe Attenuation

The probe attenuation setting (1:1 and 1:10, respectively) does not have any bearing on the actual amplifier setup, but rather tells the scope software how it must scale the measured data on the screen. A 1:10 probe reduces the signal by a factor of 10, which allows you to measure much larger signals than would be possible with a 1:1 probe.

Important Warning: While the scope can display signals between -120V and +200V with a 1:10 probe (which reduces these signals to a range between -12 and +20V), you must exercise extreme caution when working with such large signals (as a rule of thumb, anything that exceeds 20V is potentially dangerous). If you accidentally touch the source you can seriously hurt yourself (or even die). To avoid damage to yourself, the DPScope and/or your computer, always verify that the probe is truly set to 1:10 mode before contacting the circuit. Note that we decline any responsibility for damage or injury resulting from such work with high voltages – you do this at your own risk.

6.4 Horizontal



6.4.1 Scope Mode / Datalogger (Roll) Mode

In scope mode the DPScope will always acquire a full data set (200 points per channel) and store it in its internal memory before transmitting it to the computer for display. This enables very fast sample rates (up to 1 MSample/sec) because there is no limitation from the transmission speed between scope and computer. The downside is that the record length is limited to 200 points, and for very slow sample rates the time between subsequent transfers becomes larger (because it takes a while to capture 200 points at a slow rate).

Datalogger (roll) mode on the other hand is limited to relatively slow sample rates (20 samples/sec maximum, which corresponds to 0.5 sec/div), but you can have the DPScope save the captured data directly into a text file and thus have virtually unlimited storage.

6.4.2 Sample Rate

Determines the sample rate, i.e. how many times per second the signals are measured. The available range is dependent on the acquisition mode:

Normal Mode, FFT Mode: 1 sec/div to 10 μsec/div (10 S/sec to 1 MS/sec)
Equivalent Time Sampling Mode: 5 μsec/div to 0.5 μsec/div (2 MS/sec to 20 MS/sec)
Pretrigger Mode: 1 sec/div to 50 μsec/div (10 S/sec to 200 kS/sec)
Datalogger Mode: 0.5 sec/div to 1 hr/div

6.4.3 Pretrigger Mode

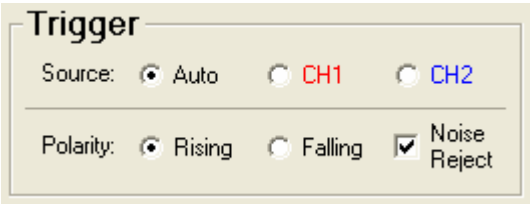
Selects between normal mode (or equivalent time sampling mode) and pretrigger mode. In Pretrigger mode you can view what happened *before* the trigger event (up to 20 divisions); the tradeoff is somewhat higher jitter, and a maximum sample rate of 200 kS/sec. For more details see the introduction.

6.4.4 Sweep Delay / Trigger Position

In normal or equivalent time acquisition mode this slider sets the amount of post-trigger delay, i.e. the time to wait after the trigger event before the sampling begins. This allows you to look at the waveform some time after the trigger with high resolution (without that feature you would have to reduce the sample rate to look at times long after the trigger, because the oscilloscope has only a limited sample memory (200 points per channel)). For repetitive signals this is effectively equivalent to a very large sample memory – if you change the slider setting the waveform will simply seem to move left or right. You can set the sweep delay to any value between 0 and 200 divisions (be careful at slow timebase settings, since during the sweep delay the DPScope will be unresponsive).

In Pretrigger Mode such a sweep delay would not make much sense (after all, you choose this mode to look at the signal before or around the trigger instant), so it is replaced by a trigger position selection; this lets you choose how much data shall be acquired after the trigger, or in other words, where the trigger position shall be within the full record. You can set the trigger position anywhere from 0 to 100% (0 to 20 divisions) in the record. To display the trigger time check then “Levels” checkbox in the Display frame.

6.5 Trigger



6.5.1 Auto / CH1 / CH2

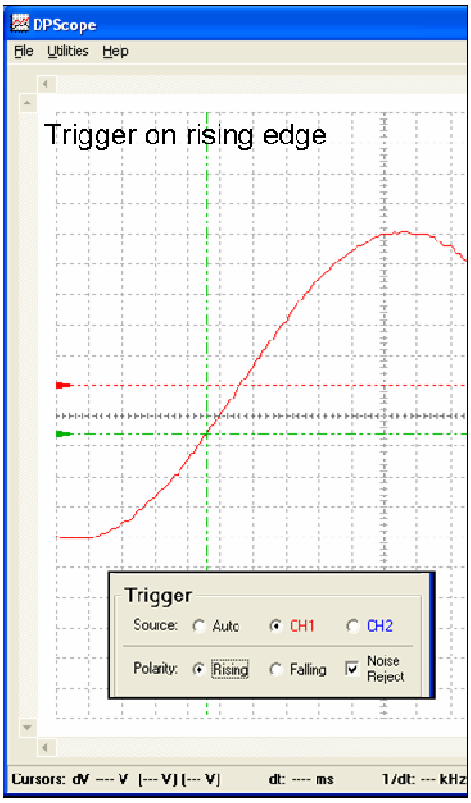
The Auto / CH1 / CH2 radio buttons let you select what the DPScope shall trigger on. A trigger is basically a condition that determines when the oscilloscope starts the data acquisition. For a repetitive signal you use this to obtain a steady picture of the signals. For a one-time signal it allows you to have the scope wait until the event of interest occurs before it starts to capture the signals.

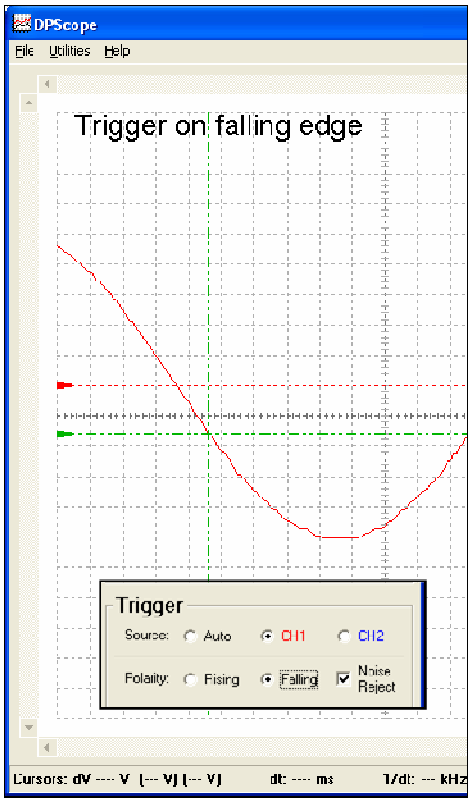
If set to “Auto” the DPScope will be free running (constantly collecting data) and not wait for any special event before capturing data. This is useful when you want to look at an unknown signal, because you will always see *something* on the screen (the scope will not wait and not display anything just because some trigger condition is not fulfilled).

If set to “CH1” or “CH2” the scope will feed the respective channel’s signal into the trigger circuitry.

6.5.2 Rising / Falling

The trigger polarity further specifies the required trigger condition. “Rising” means the signal must cross the threshold from below and rise above the threshold (i.e. a positive slope) to cause a trigger. “Falling” means the signal must cross the threshold from above (i.e. a negative slope).





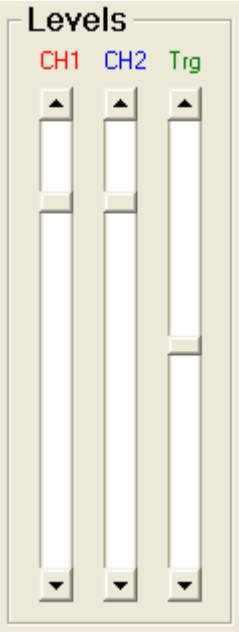
6.5.3 Noise Reject

If there is noise on the signal (and there is always some amount) this can cause false triggering. E.g. if triggering on a rising edge, the signal may actually be falling at a given instant, but a short little positive noise spike right when the signal has crossed the trigger threshold from above will look like a rising edge to the comparator and cause a trigger even though its not a true positive edge. As a result the oscilloscope will trigger on a falling edge instead of a rising edge.

The “Noise Reject” feature avoids this by requiring that the signal must remain above the threshold (in the case of a rising edge trigger; below for falling edge) for at least half a division (i.e. 5 samples). If it fails to do so (e.g. because the edge really was just a short noise spike as discussed above), the trigger event will get suppressed and the DPScope will keep waiting for a trigger.

This feature is especially valuable for slowly varying signals. Note however that if your signal period becomes shorter than about one division, activating the noise reject feature may prevent the DPScope from triggering at all. In this case either turn the feature off, or increase the sample rate so the period becomes a larger fraction of the screen width.

6.6 Levels



6.6.1 CH1 / CH2 (Channel Offsets)

Sets the channel offset for input channel 1 and 2, respectively. This lets you control where ground (zero Volts) is displayed on the vertical axis. First this expands the available voltage range for a given vertical resolution, e.g. at 0.1 V/div you could for example cover the range from 0V to + 2V, or alternatively from -2V to 0V.

A second use for the offset is to move the two waveforms apart on the screen so they are easier to discern, even though in reality they cover similar voltage ranges (e.g. 0 to 5V).

The zero position for each channel is indicated on the left side of the waveform display with a colored marker (red for CH1, blue for CH2). If “Levels” is turned on in the display control frame, two dashed horizontal lines of same colors are drawn across the display as well at that vertical position. See picture below.

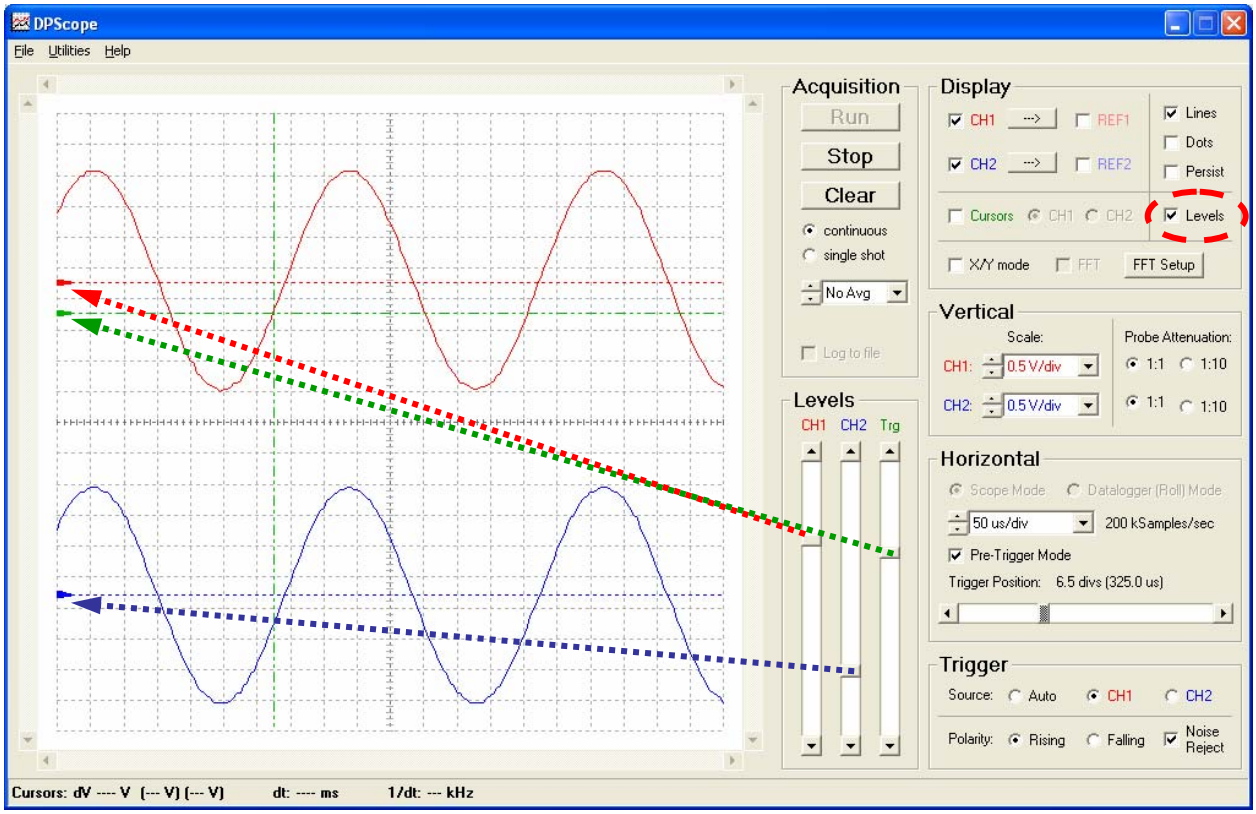
6.6.2 Trg (Trigger Level)

This slider controls the trigger threshold. If the DPScope is set up to trigger on one of the channels (CH1 or CH2), a trigger event occurs whenever the signal crosses this threshold with the desired polarity (rising or falling edge, respectively).

The zero position for each channel is indicated on the left side of the waveform display with a green marker. If “Levels” is turned on in the display control frame, a green dashed horizontal line is drawn across the display as well at that vertical position.

Usually the best approach with an unknown signal is to first display it with trigger in “Auto” mode. That way – although the waveform on the screen will not be stable – you can at least judge the voltage range that the signal spans, and set the trigger level marker somewhere within that range before setting the trigger source to the appropriate channel.

See picture below.



6.7 File Menu

6.7.1 Load Setup

Recalls a previously saved setup (all settings like acquisition mode, vertical scales, time scale, trigger setup, etc.) from a file.

6.7.2 Save Setup As

Saves the current setup (i.e. all settings like acquisition mode, vertical scales, time scale, trigger setup, etc.) to a file.

6.7.3 Export Data

Exports the currently displayed waveforms to a text file. The file contains both scope waveforms (CH1 and CH2) as well as both reference waveforms (REF1 and REF2). File format is standard CSV (comma separated values) format, which you can open and process in Microsoft Excel or any standard text editor (e.g. Notepad).

In datalogger (roll) mode it is usually more useful to select “Log to file” instead, because this will write all the data to the file, while “Export Data” only writes the latest screen interval.

6.7.4 Exit

Closes the DPScope application.

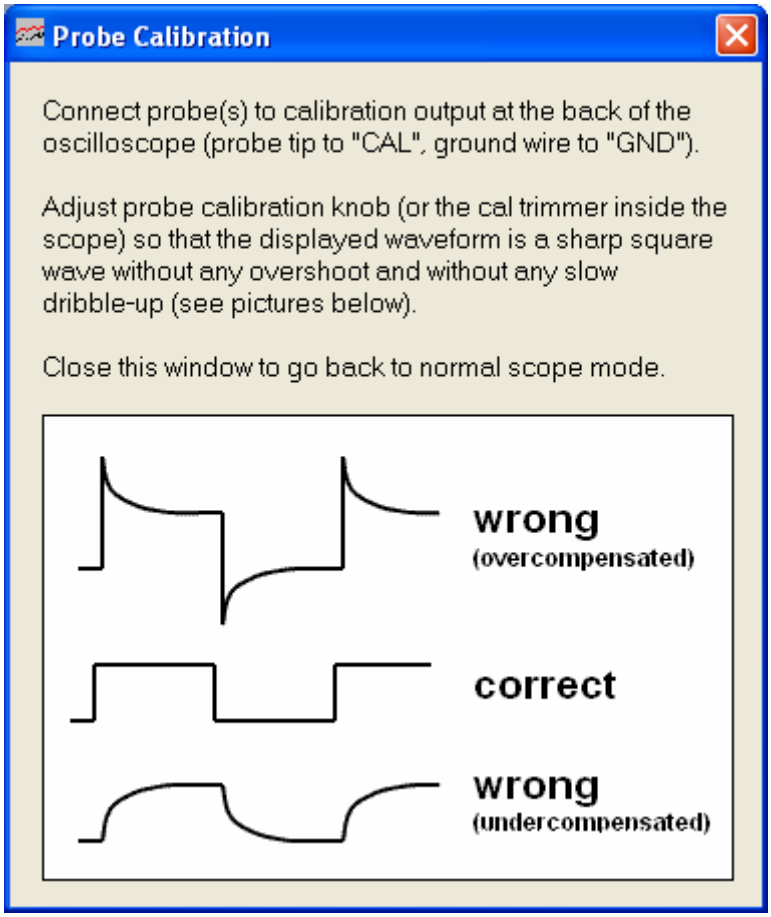
6.8 Utilities Menu

6.8.1 Probe Calibration

Selecting this menu item puts the DPScope into probe calibration mode. Acquisition will run automatically and will produce a stable picture when you probe the “CAL” output on the scope’s back panel. Depending on the probe you use (1:1 or 1:10) you may need to adjust the vertical scale and offset. No time scale adjustment is necessary (or possible); the display will automatically show roughly two periods of the calibration signal.

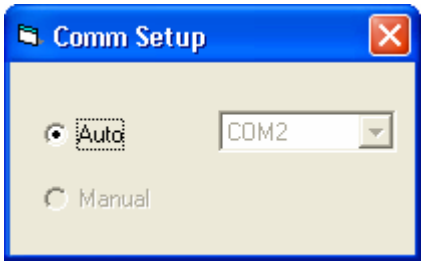
Refer to the calibration panel below as to how the waveform shall look like after calibration. On the probe there is typically a small screw (either at the probe head itself, or at a small block at the connector end) that you use to adjust the probe compensation. Since this is a trimmer

capacitor you should use a non-metal (plastic or wood) screwdriver to adjust it – a piece of metal can detune the setting when it is close, which would make it hard to adjust accurately.



6.8.2 Comm Setup

Displays the virtual COM port the DPScope is connected to. Since detection is automatic, the “Manual” setup feature is currently always grayed out.



6.9 Help Menu

6.9.1 About

Here you can find version and build of your software – please always include this information when submitting bug reports or support requests. The panel also shows the support contact info.



7 Command Interface Description

7.1 General Remarks

While the DPScope software should cover most applications, there may come a point where you would like to use the DPScope but control it from your own program. Fortunately this is rather straightforward because the DPScope's USB driver makes the interface look like a normal RS-232 serial port for the PC program. (For those interested, the interface cable contains a FT232R chip from FTDI – for more information see http://www.ftdichip.com). That means that you can use e.g. Microsoft's MSCOMM object to talk to this virtual serial port (VCP). The only difficulty is to find out which port number got assigned to the DPScope; it's easiest to bring up the DPScope software and select Utilities → Comm Setup to get this information.

The serial parameter string is "500000,8,N,1", meaning 500 kbaud, 8 data bits, no parity, one stop bit. (The speed is rather high in order to maximize the oscilloscope's acquisition speed; if that poses a problem for you, contact us and we can provide a chip with custom firmware running at any data rate you choose, e.g. 9600 baud).

The general communication scheme is as follows: The PC always initiates the communication by sending a command code (one byte) followed by none, one, or two data bytes. The DPScope processes the command and sends back an acknowledge byte and/or a block of data bytes, which indicates that it has finished processing and is ready to receive the next command. The PC must wait until it receives this response before it can send another command – the DPScope has only a very small receive buffer and we do not recommend relying on this buffer. This handshaking protocol assures that neither the PC nor the DPScope can overload the other side, and no commands can get lost.

The value of the acknowledge byte is a copy of the command code, this allows basic error detection. (That said we have tested the DPScope running many days without pause and failed to observe even a single communication error).

7.2 List of Available Commands

As mentioned above, each command code is a single byte. The list below shows the numeric value for each code, as well as an easier-to-remember name for it (you can use these names to define appropriate named constants in your control program). Each command description also lists the number of data bytes for this command (0, 1, or 2), and if the DPScope will answer with an acknowledge and/or some data bytes.

Note that all data is sent in binary format, so e.g. command code 5 is *not* the character "5" (ASCII code 53) but really binary 5. So e.g. in Visual Basic you would produce this byte as CHR(5).

To keep things simple, the lowest command codes (command code ≤ 20) are commands without additional data bytes, followed by commands with a single data byte (command code ≤ 40), and finally commands with two data bytes.

7.2.1 CMD_READADC (3):

Parameters: none

Acknowledge byte: yes

Returned data: Byte 1: signal on scope CH1
Byte 2: signal on scope CH2

This command reads back the ADC (analog-to -digital converter) for both channels directly. The resolution of the result is 8 bits. This command allows slow-speed data acquisition under full user control since it does not use the DPScope's internal sampling engine and memory.

7.2.2 CMD_PING (4):

Parameters: none

Acknowledge byte: no

Returned data: 7 bytes identifier ("DPSCOPE")

Requests echo from controller. Use this command to check if DPScope is connected and functional. If this is the case the command will return a 7-byte ASCII string (without termination character) with the contents "DPSCOPE" (without the quotation marks).

7.2.3 CMD_ABORT (6):

Parameters: none

Acknowledge byte: yes

Returned data: none

Disarms the scope (i.e. terminates a pending acquisition), so the scope is ready for a new command. Use this command in case the latest acquisition has not finished yet but you want to send a new command (e.g. change the trigger setup or the amplifier gain).

Also refer to the commands CMD_ARM and CMD_READBACK for further information.

7.2.4 CMD_READADC_10 (7):

Parameters: none

Acknowledge byte: yes

Returned data: Byte 1: signal on scope CH1 (MSB)
Byte 2: signal on scope CH1 (LSB)
Byte 3: signal on scope CH2 (MSB)
Byte 4: signal on scope CH2 (LSB)

This command reads back the ADC (analog-to-digital converter) for both channels directly. The resolution of the result is 10 bits. This command allows slow-speed data acquisition under full user control since it does not use the DPScope's internal sampling engine and memory.

The 10-bit measurement result can be aligned to the left or to the right of the 16-bit data word of each channel; refer to CMD_ADCON_FORM for details.

7.2.5 CMD_TRIG_SOURCE (21):

Parameters: Byte 1: Trigger source

Acknowledge byte: yes

Returned data: none

Selects the trigger source: auto = 0, CH1 = 1, CH2 = 2

7.2.6 CMD_TRIG_POL (22):

Parameters: Byte 1: Trigger polarity

Acknowledge byte: yes

Returned data: none

Selects the trigger polarity: rising edge = 0, falling edge = 1

7.2.7 CMD_READBACK (23):

Parameters: Byte 1: number of bytes (N) *per channel* to send

Acknowledge byte: yes

Returned data: 2*N bytes of captured signal data

Initiates readback of acquired data record. Note that the command parameter defines the number of bytes *per channel*, not the total number of bytes. The maximum is 205.

The data format depends which type of acquisition has been performed. In the case of a CMD_ARM the data for the two channels is interleaved, i.e. the sequence is

CH1 value 1
CH2 value 1
CH1 value 2
CH2 value 2
…….
…….
CH1 value N
CH2 value N

For a preceding CMD_ARM_FFT (which acquires only a single channel, but with twice the number of points) the returned data set is for this particular channel, e.g. for channel 1:

CH1 value 1
CH1 value 2
…….
…….
CH1 value 2*N

Data gets sent at maximum speed (500 kbaud), so you need to make sure that your program can receive data fast enough or that your receiver has a sufficiently large buffer (up to 410 bytes). If your control program runs on a PC this is not an issue because the FTDI interface driver takes care of the buffering, but if your controller is e.g. another microcontroller this may be too fast. If that is the case then contact us and we can provide a custom firmware chip that uses a slower data rate (e.g. 9600 baud).

7.2.8 CMD_SAMPLE_RATE (24):

Parameters: Byte 1: sample rate code

Acknowledge byte: yes

Returned data: none

Sets the sample clock frequency. The following sample rate codes are available:

SAMPLE_RATE_20M_ET = 0
SAMPLE_RATE_10M_ET = 1
SAMPLE_RATE_5M_ET = 2
SAMPLE_RATE_2M_ET = 3
SAMPLE_RATE_1M = 4
SAMPLE_RATE_500k = 5
SAMPLE_RATE_200k = 6
SAMPLE_RATE_100k = 7
SAMPLE_RATE_50k = 8
SAMPLE_RATE_20k = 9
SAMPLE_RATE_10k = 10
SAMPLE_RATE_5k = 11
SAMPLE_RATE_2k = 12

SAMPLE_RATE_1k = 13
SAMPLE_RATE_500 = 14
SAMPLE_RATE_200 = 15
SAMPLE_RATE_100 = 16
SAMPLE_RATE_50 = 17
SAMPLE_RATE_20 = 18
SAMPLE_RATE_10 = 19

Note that the equivalent time sampling rate codes (0 to 3) have the same effect as code 4 – in all cases the oscilloscope will sample at 1 MSample/sec. To achieve equivalent time sampling you will need to repeat the acquisition, each time stepping the fine delay (see CMD_ARM for details) to its next appropriate value.

7.2.9 CMD_NOISE_REJECT (25):

Parameters: Byte 1: noise reject off (0) or on (1)

Acknowledge byte: yes

Returned data: none

Turns the trigger noise reject on or off. If noise reject is turned on then the trigger level needs to stay continuously above (for rising edge trigger) or below (for falling edge trigger) the trigger threshold for at least half a horizontal division (i.e. 5 sample periods). This avoids false triggering in the presence of noise on the signal, which is especially of concern for signals with slow slew rates.

7.2.10 CMD_ARM (26):

Parameters: Byte 1: fine delay in 0.5us steps (0…19)

Acknowledge byte: yes

Returned data: none

This command initiates an acquisition and sets fine delay for equivalent time sampling (in real-time sampling mode or pretrigger mode just set the fine delay to zero). If the trigger mode is “auto” then the acquisition will start right after this command. Otherwise the scope will wait until a trigger event on the selected channel (CH1 or CH2) arrives and only then acquire signal data. You can query the status of the acquisition (still waiting/sampling, or acquisition complete) with the CMD_READBACK command.

Note: Do not send any commands while scope is armed. It may be busy sampling just when you send a command, resulting in loss of the command and possible communication lockup. Instead first send a CMD_ABORT command and wait for the respective acknowledge byte. The scope has a small receive buffer so a single CMD_ABORT will not get lost even of the scope is currently busy otherwise.

7.2.11 CMD_ADCON_FORM (27):

Parameters: Byte 1: resolution mode (high = 0, low = 1)

Acknowledge byte: yes

Returned data: none

Selects whether the scope shall store the upper 8 bytes (i.e. low resolution) or the lower 8 bytes (high resolution) of the 10-bit ADC result (due to speed and memory restrictions the scope stores only 8 bits, while the ADC really has 10 bit resolution). *Normally the parameter will be 1* so you can acquire the full available voltage range. Mode 0 on the other hand allows digitizing the lower quarter of the range with four times higher resolution. Note that in this case the value will wrap around when you exceed this partial range, i.e. for example an input signal of 0.35 (1/4 + 0.1) of full range will result in the same stored value as a signal of 0.1

7.2.12 CMD_CAL_MODE (28):

Parameters: Byte 1: probe calibration mode (off = 0, on = 1)

Acknowledge byte: yes

Returned data: none

Turns the probe calibration mode on or off.

7.2.13 CMD_PRETRIGGER_MODE (29):

Parameters: Byte 1: pretrigger mode (disabled = 0, enabled = 1)

Acknowledge byte: yes

Returned data: none

Enables and disables pretrigger mode. If you enable pretrigger mode you also need to issue a CMD_POST_TRIG_CNT to specify the amount of post-trigger data to acquire.

7.2.14 CMD_TIMER_PRESCALE (30):

Parameters: Byte 1: time prescaler code

Acknowledge byte: yes

Returned data: none

Sets the timer prescaler for pretrigger mode. In pretrigger mode the sample timing is generated through a timer interrupt at regular intervals. The prescaler in combination with the timer period value (see CMD_TIMER_PERIOD) define the actual sample period.

The prescaler code has the following meaning:

Code = 0 → prescaler = 1
Code = 1 → prescaler = 8
Code = 2 → prescaler = 64
Code = 3 → prescaler = 256

You can calculate the sample interval as follows:

Sample_interval in μs = (timer_period_value / 32) * prescaler

The shortest functional interval is 5us (i.e. 200 kSamples/sec), corresponding to a timer_period_value of 160 and prescaler = 1 (i.e. prescaler code = 0).

So for e.g. 100 samples per second (10 ms = 10000 μs sample interval) you would set prescaler = 64 (prescaler code = 2) and timer_period_value = 1250.

7.2.15 CMD_POST_TRIG_CNT (31):

Parameters: Byte 1: post-trigger sample count

Acknowledge byte: yes

Returned data: none

This command only needs to be issued in pretrigger mode. It defines the number of samples (per channel) to acquire *after* the trigger event. Reasonable values are 0 to the maximum buffer size (205). If you want to have 100% pretrigger, set it to 0. For 50% pretrigger set it to 100 (since in the standard DPScope software only 201 samples are displayed).

7.2.16 CMD_SERIAL_TX (32):

Parameters: Byte 1: data byte to send

Acknowledge byte: yes

Returned data: none

Transmits one byte of data through the CAL_OUT pin at 9600 baud, 8 data bits, no parity, 1 stop bit. You can use this to control some other peripheral connected to CAL_OUT and GND.

7.2.17 CMD_STATUS_LED (33):

Parameters: Byte 1: LED state (0 = off, 1 = on)

Acknowledge byte: yes

Returned data: none

Turns the status LED on the front panel on or off, respectively. This is a quick and easy visual check whether you can successfully send data to the DPScope.

7.2.18 CMD_TRIG_LEVEL (41):

Parameters: Byte 1: trigger level MSB
Byte 2: trigger level LSB

Acknowledge byte: yes

Returned data: none

Sets the trigger level (0...1023), same value for both channels. The trigger level value is (256*MSB + LSB). The level range 0…1023 corresponds to the full display scale at the given voltage scale (i.e. at 1 V/div it spans 20V). Which absolute voltage this corresponds to depends on the voltage offset and the selected gain of the trigger channel.

7.2.19 CMD_PRE_GAIN (42):

Parameters: Byte 1: channel (1 = CH1, 2 = CH2)
Byte 2: pre-amplifier gain (0 = gain 1, 1 = gain 10)

Acknowledge byte: yes

Returned data: none

Selects pre-amp gain (1 or 10) for the particular channel. The total gain is the product of preamp gain and programmable-gain-amplifier (PGA) gain:

Total_gain = preamp_gain * PGA_gain

A total gain corresponds to a full-scale voltage range (ADC values spanning 0 to 256 in low-resolution mode) of 20 V.

7.2.20 CMD_GAIN (43):

Parameters: Byte 1: channel (1 = CH1, 2 = CH2)

Byte 2: gain code

Acknowledge byte: yes

Returned data: none

Selects programmable-gain amplifier's (PGA) gain for the particular channel. The gain code has the following meaning:

Gain code = 0 → PGA_gain = 1
Gain code = 1 → PGA_gain = 2
Gain code = 2 → PGA_gain = 4
Gain code = 3 → PGA_gain = 5
Gain code = 4 → PGA_gain = 8
Gain code = 5 → PGA_gain = 10
Gain code = 6 → PGA_gain = 16
Gain code = 7 → PGA_gain = 32

Note that gains higher than 10 (i.e. 16 and 32) should be avoided because they may result in excessive noise and/or oscillations of the input amplifier.

The total gain is the product of preamp gain and programmable-gain-amplifier (PGA) gain:

Total_gain = preamp_gain * PGA_gain

A total gain corresponds to a full-scale voltage range (ADC values spanning 0 to 256 in low-resolution mode) of 20 V.

7.2.21 CMD_SET_DAC (44):

Parameters: Byte 1: DAC channel, enable bit, and DAC value MSB
Byte 2: DAC value LSB

Acknowledge byte: yes

Returned data: none

This command controls the DPScope's DAC which produces the channel offset voltages. The syntax is a bit hardware oriented (refer to the Microchip MCP4822 data sheet for the protocol); the two data bytes hold DAC channel, enable bit, and DAC value. The enable bit is always on. The DAC value has 12 bits, thus can range from 0 to 4095; the value corresponds to the DAC output voltage in mV. Note that due to the DPScope's 1:4 input attenuator the effective offset is three times the DAC voltage, e.g. a DAC value of 1000 corresponds to a channel offset of 3000mV = 3V.

To build the two-byte data word, use the following formulas (in C syntax):

CH1: Byte1 = 0x90 + (DAC_value >> 8)
Byte2 = DAC_value & 0xff;

CH2: Byte1 = 0x10 + (DAC_value >> 8)
Byte2 = DAC_value & 0xff;

In Visual Basic this looks as follows

CH1: Byte1 = &H90 + (DAC_value \ 256)
Byte2 = DAC_value Mod 256;

CH2: Byte1 = &H10 + (DAC_value \ 256)
Byte2 = DAC_value Mod 256;

7.2.22 CMD_ARM_FFT (45):

Parameters: Byte 1: fine delay in 0.5us steps (0…19)
Byte 2: channel to acquire (1 = CH1, 2 = CH2)

Acknowledge byte: yes

Returned data: none

This command initiates an acquisition and sets fine delay for equivalent time sampling (in real-time sampling mode or pretrigger mode just set the fine delay to zero). This command works like CMD_ARM with the exception that it acquires only one channel but twice as many data points (410) on this channel. The DPScope software uses it in FFT mode (hence the name of the command), but you can employ it just as well for “normal” acquisition if you need more than 205 data points.

If the trigger mode is “auto” then the acquisition will start right after this command. Otherwise the scope will wait until a trigger event on the selected channel (CH1 or CH2) arrives and only then acquire signal data. Note that the trigger channel does not need to be the channel you want to acquire. You can query the status of the acquisition (still waiting/sampling, or acquisition complete) with the CMD_READBACK command.

Note: Do not send any commands while scope is armed. It may be busy sampling just when you send a command, resulting in loss of the command and possible communication lockup. Instead first send a CMD_ABORT command and wait for the respective acknowledge byte. The scope has a small receive buffer so a single CMD_ABORT will not get lost even of the scope is currently busy otherwise.

7.2.23 CMD_SET_DELAY (49):

Parameters: Byte 1: delay value MSB
Byte 2: delay value LSB

Acknowledge byte: yes

Returned data: none

Sets the post-trigger delay (only used in normal and equivalent time acquisition mode).

The delay value is (256 *MSB + LSB).

The delay value is expressed in sample periods so the absolute delay time depends on the currently selected sample rate. Be careful when selecting large delay values at slow sample rates – the scope will be unresponsive during the delay time. Also note that in equivalent time sampling mode (modes 0 to 3) the actual sampling rate is 1 MS/sec, so the delay value is always in 1us increments.

7.2.24 CMD_TIMER_PERIOD (51):

Parameters: Byte 1: timer period value MSB
Byte 2: timer period value LSB

Acknowledge byte: yes

Returned data: none

Sets the timer period value for pretrigger mode. In pretrigger mode the sample timing is generated through a timer interrupt at regular intervals. The prescaler (CMD_TIMER_PRESCALE) in combination with the timer period value define the actual sample period.

The timer period value is (256 *MSB + LSB).

You can calculate the sample interval as follows:

Sample_interval in μs = (timer_period_value / 32) * prescaler

The shortest functional interval is 5us (i.e. 200 kSamples/sec), corresponding to a timer_period_value of 160 and prescaler = 1 (i.e. prescaler code = 0).

So for e.g. 100 samples per second (10msec = 10000 μs sample interval) you would set prescaler = 64 (prescaler code = 2) and timer_period_value = 1250.