LabNation SmartScope

User Manual

V1.1.6

2017/09/19

2017/03/04 - Pages have been updated to reflect new features in SmartScope app v0.12.0.0

2017/04/26 - Correction by Riemer: Digital Waveform Generator

2017/06/21 - Correction by Riemer: WiFi Bridge

2017/06/30 - Correction by Riemer: Recording Data to Disk

2017/07/15 - Add page by Riemer: ADC resolution - voltage dividing/multiplying stage

2017/07/30 - Correction by Riemer: ADC resolution - voltage dividing/multiplying stage

2017/08/21 - Add page by Riemer: Measurements (v0.13.0.0)

2017/09/16 - Edit page by Riemer: Measurements (v0.13.0.0)

2017/09/16 - Edit page by Riemer: Main Page (v0.13.0.0)

Editado de la WIKI de LabNation: http://wiki.lab-nation.com/index.php/Main_Page

Es una edición editada del Manual del Usuario en HTML que se encuentra en la WEB del fabricante del SmartScope.

SmartScope User Manual

Main Page

Contents

Demonstration mode

All of our software installations contain a **demonstration mode**, which is automatically activated in case no SmartScope is connected. This allows everyone to **evaluate the software**. Simply grab our installer from our download page and give it a spin!
For Android phones and tablet we have implemented an **Audio Scope**, which allows you to test our software on real-world signals.

Installing the software and connecting the SmartScope

- Connecting to a SmartScope over the network
- Connecting to a SmartScope with a USB cable
    - Connecting on Windows
    - Connecting on macOS
    - Connecting on Linux
    - Connecting on Android
    - Connecting on iOS

SmartScope User Manual

(2017/09/16 - All pages have been updated to reflect the new changes in v0.13.0)

- **SmartScope basics**
    - Probes: x1 or x10 modus
    - Howto Videos
- **Core functionality**
    - Oscilloscope functionality
    - Logic Analyzer functionality
    - Mixed mode
    - Panorama (RAM zoom) functionality
    - Cursors
    - Measurements
    - Reference waves
    - Arbitrary Waveform Generator (AWG)
    - Digital Waveform Generator
    - Recording data to disk
- **Extended functionality**
    - Advanced triggering options
    - Using the Operators
    - Using the Protocol Decoders
    - FFT
    - XY Mode
    - High speed signals - Peak Detect Acquisition - Ecquivalent Time Sampling
- **Global/basic functionality**

- Changing the appearance of the app
- Keyboard shortcuts
- Main menu
- Cue card
- Left-handed mouse patch

- **Customizing/extending the functionality of your SmartScope**
    - Creating your own Operator
    - Creating your own Protocol Decoder
    - Controlling your SmartScope from Matlab
    - Controlling your SmartScope from LabView

- **Using the smartscope over a network connection**
    - Using the smartscope on the network

In case of a crash: please send in the CrashReport!

We're not proud of crashes. On the contrary, we do everything possible to fix and avoid them. Please help us and fellow users by sending in the crash report, automatically generated in the unfortunate event of a crash.

- Learn about Crash Reports

Hardware

- Hardware specs
- ADC resolution - voltage dividing/multiplying stage
- Using the micro USB connector
- Connectors pinout
- Probe calibration

General

- Changelog
- Suggestion box
- Host system requirements
- User Support Forum
- Lab-Nation main page

Sources

- Sources

Other

- Sandbox

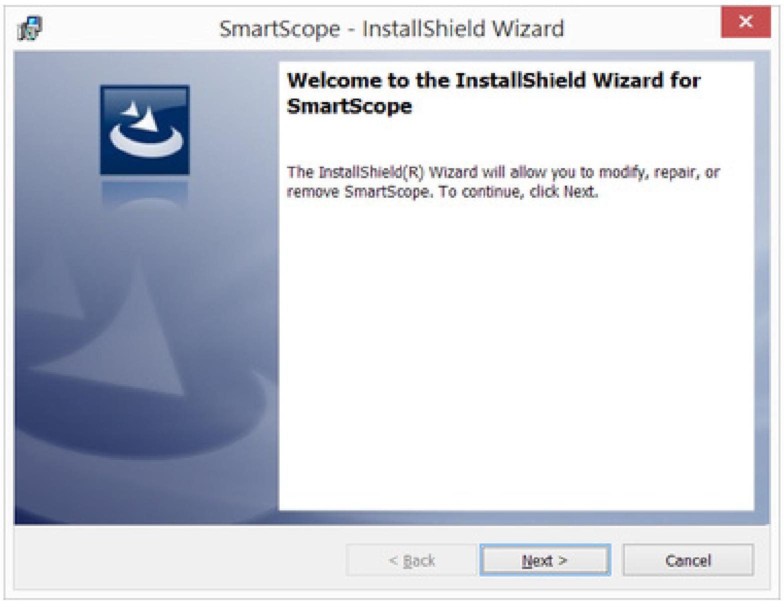
This page was last modified on 16 September 2017, at 23:18.

Getting started on Windows

Contents

Downloading and installing the SmartScope software on Windows

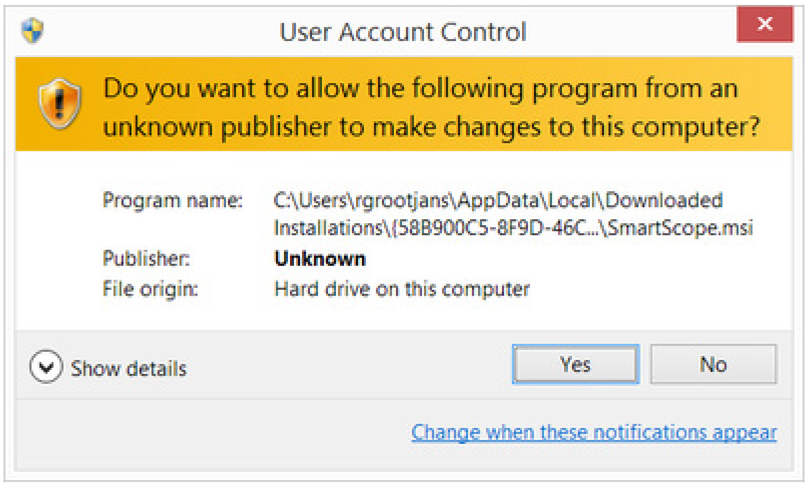
1. Go to our main download page and download our application for Windows.
2. Once the download is complete, run the file you just finished downloading. (note: since we haven't yet signed our program using a certificate, you might need to navigate through some warnings)
    1. Welcome screen: sit tight and click next


SmartScope - InstallShield Wizard

Welcome to the InstallShield Wizard for SmartScope

The InstallShield(R) Wizard will allow you to modify, repair, or remove SmartScope. To continue, click Next.

    2. Administration elevation: we've fully automated the process of driver installation for you, and as with any driver installation this requires administrator rights.
    Hence, click OK to proceed.


User Account Control

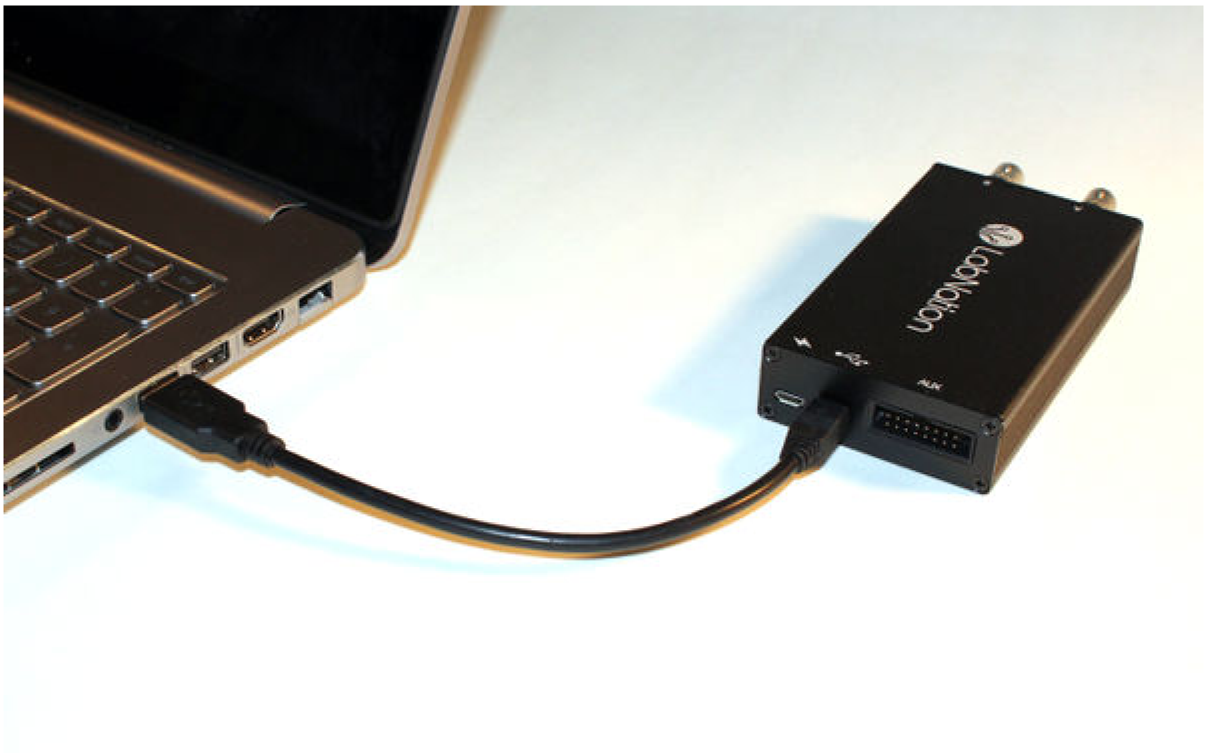
Do you want to allow the following program from an unknown publisher to make changes to this computer?

Program name: C:\Users\rgrootjans\AppData\Local\Downloaded Installations\{58B900C5-8F9D-46C...\SmartScope.msi
Publisher: **Unknown**
File origin: Hard drive on this computer

    3. That's all folks!
    Click Finish to exit the installation program.

Connecting the SmartScope to your Windows PC

1. Connect the SmartScope's central miniB USB port with a miniB cable to your PC, as shown in the image below:



2. Next, simply start the SmartScope application through any of these means:

   - Find the SmartScope icon on your desktop and give it a double-click
   - Hit the Start button on your keyboard, type "SmartScope" and hit enter
   - Browse to C:\Program Files (x86)\LabNation\SmartScope and double-click on the SmartScope.exe file

Additional step for Windows XP

On most Windows XP systems, the driver installation will not have the intended result. In this case, no SmartScope will be detected. To solve this, download and install the Zadig driver. Make sure you use the 'Zadig for Windows XP' link. In the app, ensure SmartScope is selected and hit the large 'install driver' button. After that, the SmartScope will be detected correctly in the SmartScope app.

Troubleshooting connectivity issues on Windows

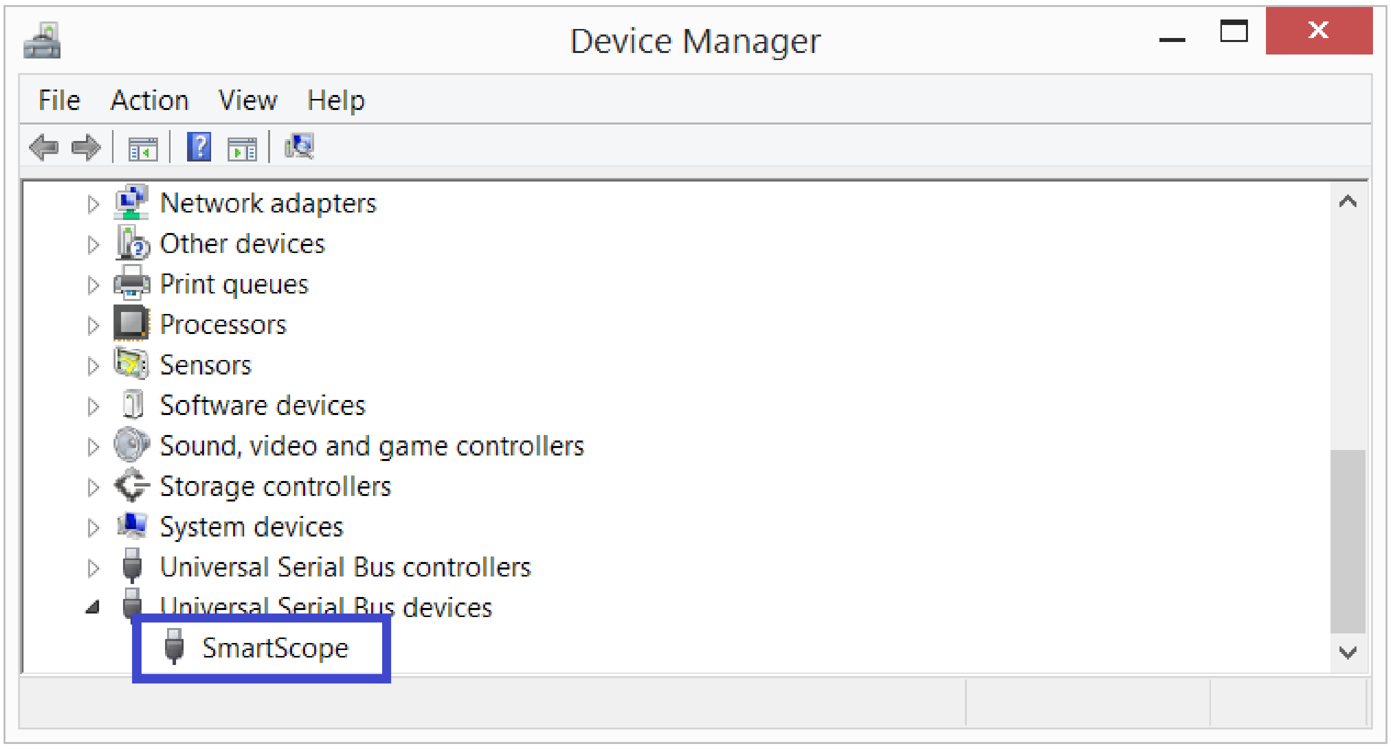
Whenever in doubt, please go through the steps below and let us know in the forum where things go wrong.

Trivial suggestions

- Make sure you're using the miniB port as shown in the image above, and not the microB port on the side
- Try out whether another USB cable might help
- Try out different USB ports on the PC
- Try out with a different PC/host

More advanced steps

- Download and install the latest software from the LabNation download page
- In the main menu, select System -> Install driver, and restart the application. Make sure you have administrator rights (which you need to install any new hardware)
- Open up the Device Manager, eg as explained on this page, and check whether the SmartScope is shown as in the image below



Getting help

If these steps didn't solve your problems, by all means let us know at the LabNation forum. Please make sure you include in your post:

- that you've gone through the steps above
- whether and how the SmartScope is showing up in the Device Manager
- if you hear a USB connection sound when you plug/unplug the SmartScope

Getting started on Android

Contents

Compatibility

In 2016, it is quite fair to assume any new Android phone/tablet is supporting USB On-The-Go. Nevertheless, on the following page you can find more information to figure out whether the SmartScope will work on your device.

- Is my Android device compatible?

Download and install the SmartScope app

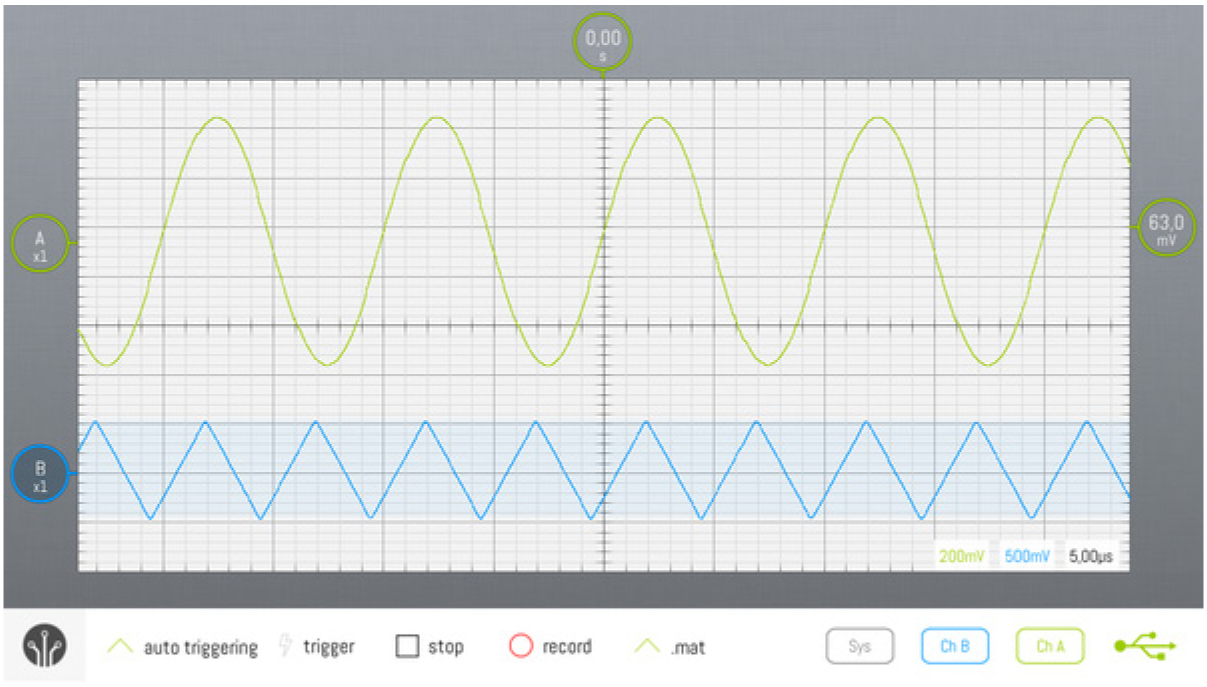
The app is available for free through the Play store:

- Link to the SmartScope app in the Play store

Connecting the SmartScope to your Android device

1. Connect the central miniB port of the SmartScope using a regular miniB USB cable
2. Connect this USB cable to your Android device using an OTG cable. Sources: Lab-Nation

1. Start the SmartScope app, and go solve your problems!!



In case nothing happens when you connect the SmartScope

In certain versions of Android, you might have to let the phone identify the smartscope:

- Settings -> Device Connections -> Identify USB-device
- There you get to confirm that you want to use the Smartscope app for this USB device.

SmartScope basics

Probes: x1 or x10 modus

First the specs you'll get using the probes at 1X/10X with SmartScope (they are not exact, just approximate reference values):

	x1	x10
Impedance	1 MΩ // 100pF	10 MΩ // 20pF
Bandwidth	6 MHz	30 MHz
Max. Voltage Range	±30 V	±300 V
Max. Amplitude	40Vpp	400Vpp
Voltage Resolution (V/AdcUnit)	2 mV	20 mV

Let's talk about all that...

Impedance: This will be the load you'll apply to the point under test. When you are probing a node you are adding a 1/10MΩ resistor to ground in that point and a parallel capacitor of 100/20pF to ground too. That means that a 10X probe will be better because it'll have less impact on the actual signal in the circuit.

Bandwidth: To put it simply, the bandwidth tells you from what frequency up the signals will be 'smoothed'. If you are working with sine waves you'll only get attenuation, but for square or any complex signal they need to be 1/10 of the bandwidth or slower, otherwise you'll see distorted waveforms. So what this means? At 1X you'll be limited to 600Khz square signals but at 10X you can go up to 3Mhz square signals without significant deformation. So once again a 10X probe will be better. The probe is actually 60Mhz but the scope has 30Mhz bandwidth (You actually always want your probe to be of higher bandwidth that the scope otherwise you'll get double attenuation).

By the way, all this does not mean that you cannot see signals faster than 3Mhz, you can, actually you can see signals all the way to 50Mhz (half of the sampling frequency), but after 3Mhz you'll start to notice the loss of higher frequency components, meaning edges will not look as sharp as they actually are.Around 20Mhz or so you'll see everything like a sine wave, and after that you'll get into aliasing problems reaching the limits of the sampling rate.

Max. Voltage Range: Since you have 10x attenuation you'll have 10x more range. So 10x probe is better again. Actually, it is a bit more complex, because the probe could be damaged if the voltage and the frequency are both high (and it could also damage the scope). Without entering in details this is a simple table of the safe voltage ranges with the probe in 10x:

Frequency	Voltage Range
0 - 20 kHz	±200 V / 120 VAC
20 kHz - 200 kHz	±100 V / 60 VAC
200 kHz - 50 MHz	±25 V / 16 VAC

Max. Voltage Resolution: This is the minimal voltage difference that the ADC of the scope can measure (with the volts/div in the lowest ranges of course). Here 1x probe is better because due to the probe attenuation you'll have 1/10 of the resolution. But of course this will only be true in the lowest ranges, once you get to 500mv/div you'll have the same resolution using 1x or 10X probe.

So, long story short, *a 10x probe will be better* in all cases except two:

- When you want to measure small signals (under 2Vpp) with more resolution than 20mV.
- When you are measuring signals of 1Mhz or lower and you want to filter out high frequency noise (over 6Mhz).

Howto Videos

Contents

1 SmartScope Controls

https://youtu.be/feGdiQOJWHI

2 Acquisition Overview

https://youtu.be/7oYi9ylijAc

2.1 (partial) transcript

The system measurement box shows the length of the acquisition buffer as well as the sample rate. Right now that is 2.62ms and 100MHz. Given the memory of 4M samples, we can acquire up to 40ms without subsampling (4M samples / 100M samples per second). However this is not enabled by default as it would slow down the data refresh rate. By default the acquisition buffer is twice the size of what you see on the grid. I open the acquisition overview buffer by pressing V. The viewport is moved around by dragging the highlighted part. Moving the trigger is done by dragging the shaded part...

3 Cursors and measurements

https://youtu.be/rYmdZJyOu0Y

4 Acquisition modes

https://youtu.be/WKrIH7zo8M0

This page was last modified on 27 January 2016, at 09:54.

Core Functionality

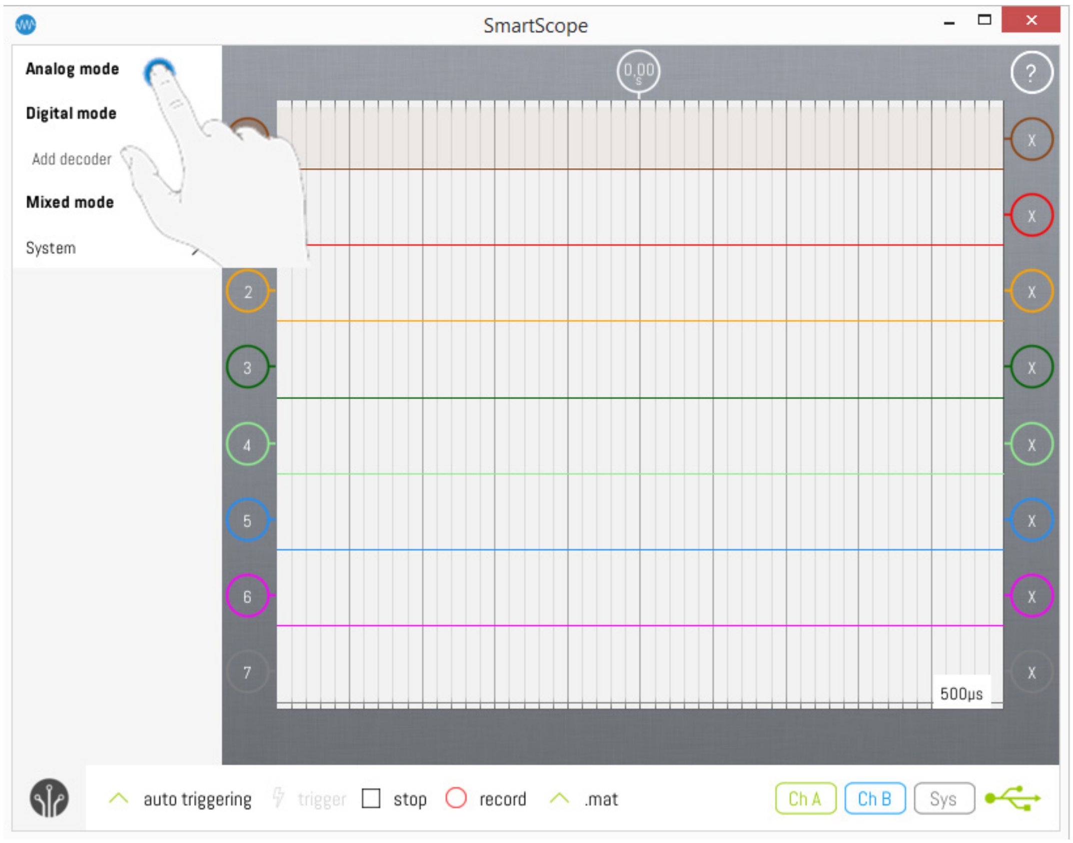
Oscilloscope functionality

One of the main features of the SmartScope is its dual channel 100MS/s oscilloscope functionality. Where a logic analyzer allows you only to detect whether a positive signal is above or below a certain voltage, an oscilloscope will give you the actual voltage for every sample. This section describes how to operate the SmartScope software to effectively use the oscilloscope functionality.

Contents

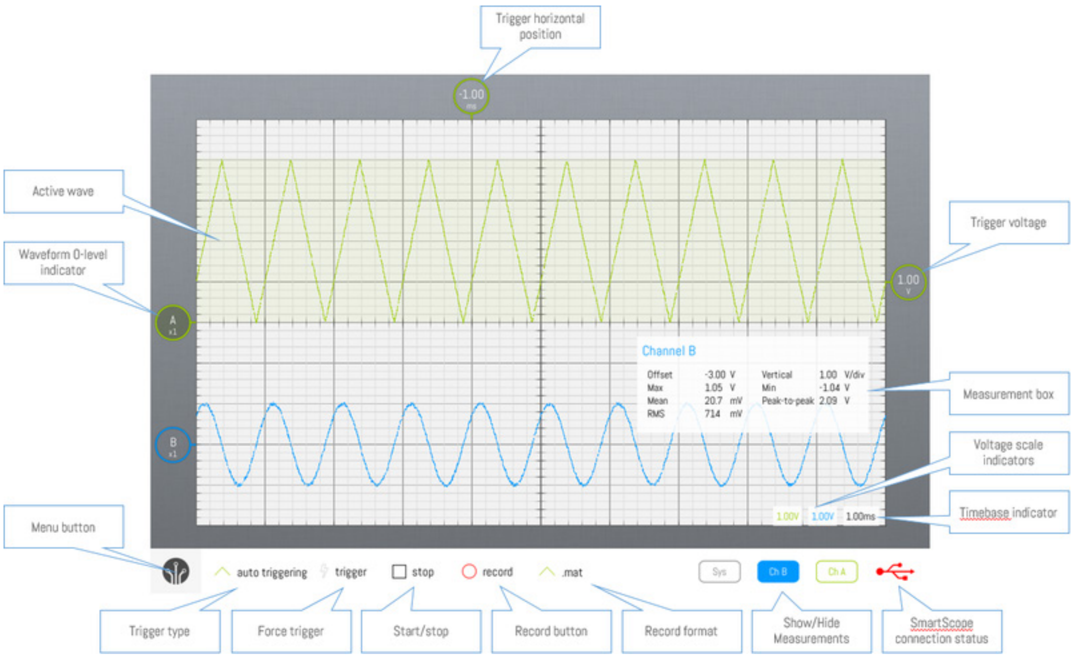
1 Selecting the Oscilloscope mode

By default, upon startup the SmartScope app will be in Analog mode. Whenever you are in a different mode, and want to return to the oscilloscope mode, simply select it at the top of the main menu:



2 Oscilloscope main screen

The key parts of the main screen are shown on the image below. This image can be used as reference for the rest of this text.



Channel B

Offset	-3.00 V	Vertical	1.00 V/div
Max	1.05 V	Min	-1.04 V
Mean	20.7 mV	Peak-to-peak	2.09 V
RMS	714 mV		

3 Visualizing the signal

When applying a signal to the inputs of the oscilloscope connectors, you will want the signal to be displayed nicely on the screen. Since the oscilloscope has pretty large input range, this will involve some scaling and panning depending on the signal. Usually, this involves zooming out both in voltage (vertically) and on the timebase (horizontally) until you're happy with the size of the signal, and then panning the wave to the area where you want it to be.

3.1 Selecting the active wave

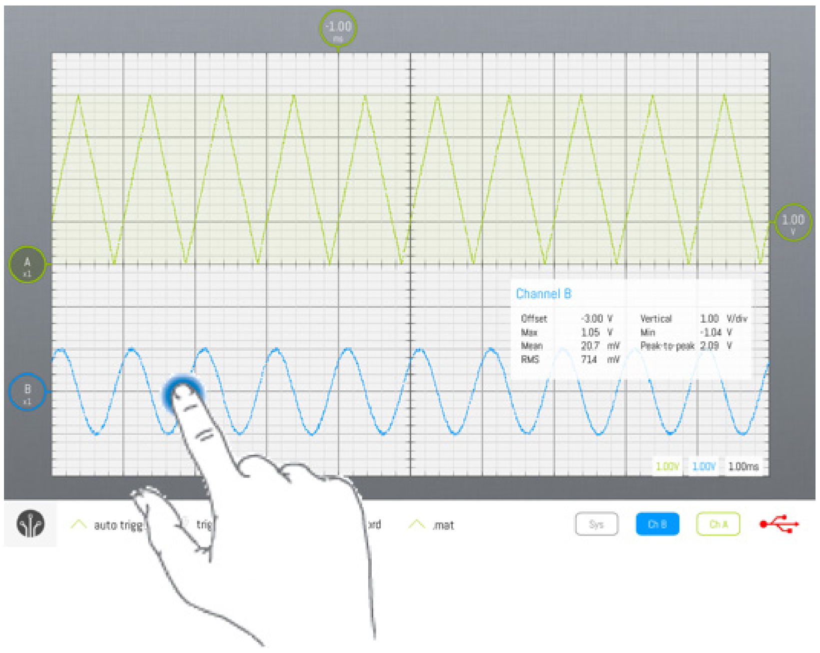
Scaling and panning only affect the 'active wave' (the signal currently selected). You can easily see which wave is currently highlighted, as:

- Its background is slightly highlighted
- Its 0-level indicator has a darker color

Touch	Mouse	Keyboard
Tap	Click	(Shift+) Tab

Changing active wave can be done in the following ways

See the image below on how to select the blue wave as active wave:



Channel B

Offset	-3.00 V	Vertical	1.00 V/div
Max	1.05 V	Min	-1.04 V
Mean	20.7 mV	Peak-to-peak	2.09 V
RMS	714 mV		

Alternative ways where the active wave will be selected automatically:

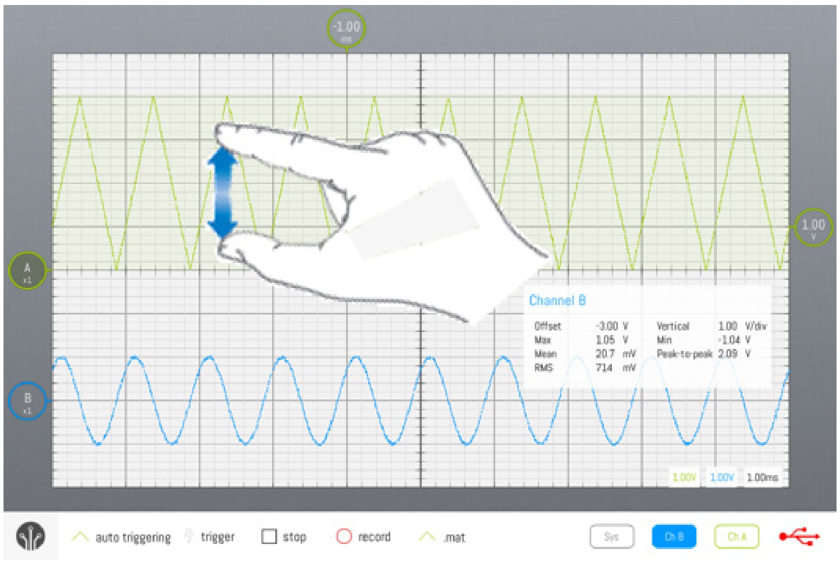
- When you start draggin (by touch or mouse) a non-active wave, it becomes active
- When you start pinching a non-active wave, it becomes active

3.2 Voltage scaling

Zooming in or out the voltage scale of the active wave is done through the following operations:

Touch	Mouse	Keyboard
Vertical Pinch Drag pickingwheel	Shift+MouseWheel Drag pickingwheel	PgUp or PgDown

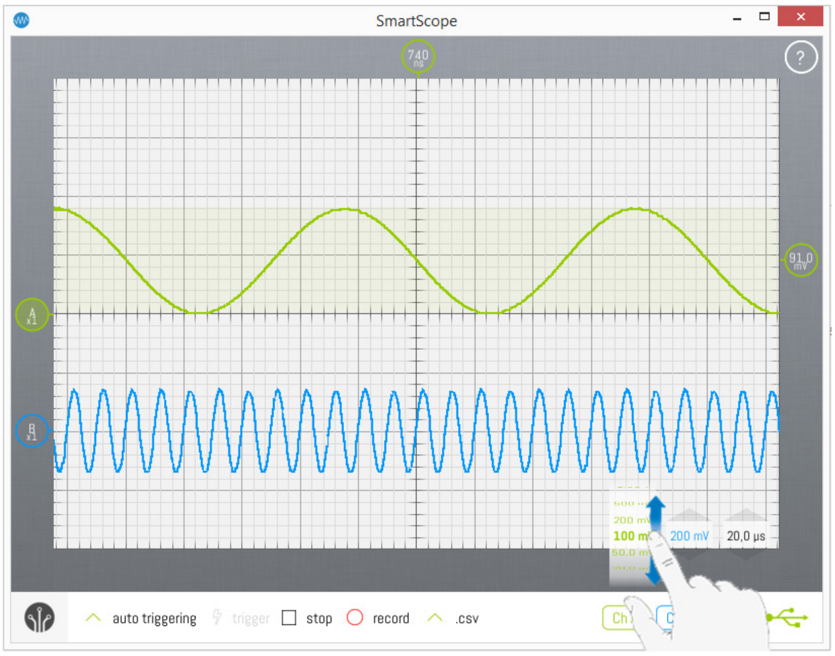
The grid in the background can help you determine the voltage level or amplitude of your signal. The grid is split in 10 divisions, and the voltage of each division is displayed in the Voltage division indicators, on the bottom-right of the screen. Notice that each signal can have a different scaling factor, and such there is one indicator for each wave. The image below shows how touch-based voltage zooming on the green wave can be done:



Channel B

Offset	-3.00	V	Vertical	1.00	V/div
Max	1.05	V	Min	-1.04	V
Mean	20.7	mV	Peak-to-peak	2.09	V
RMS	714	mV			

Additionally, the voltage can also be adjusted using the picking wheels at the bottom-right of the screen. To do so, simply drag them up or down as shown in the image below.

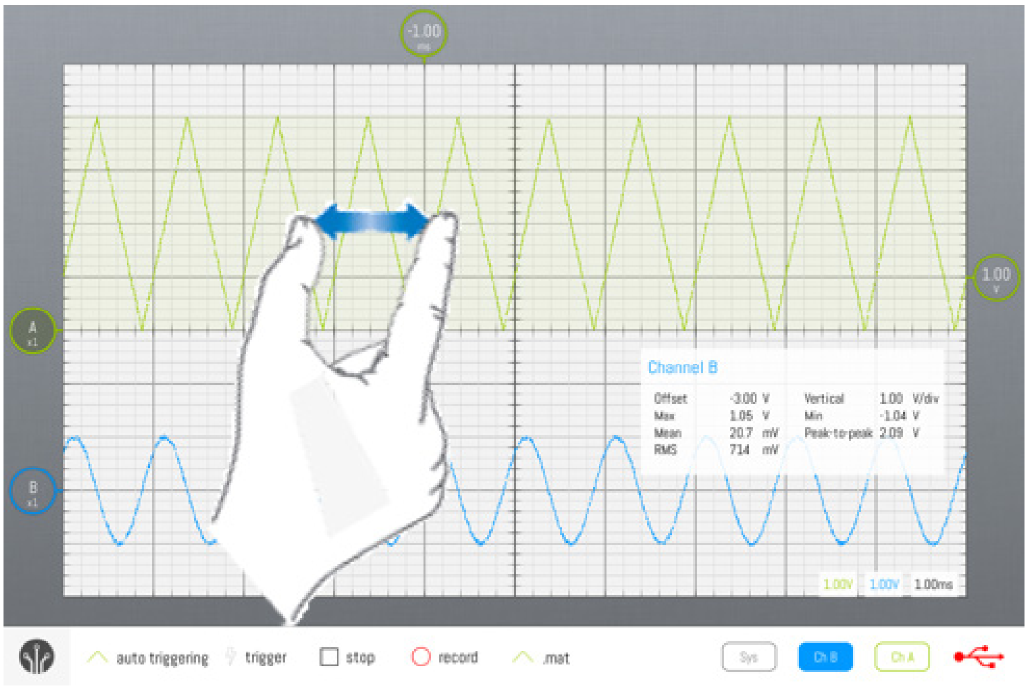


3.3 Timebase scaling

Depending on whether you want to visualize a fast signal or a slow signal, you'll want to set the timebase accordingly. This can be achieved using the following operations:

Touch	Mouse	Keyboard
Horizontal Pinch Drag pickingwheel	Mousewheel Drag pickingwheel	Home or End

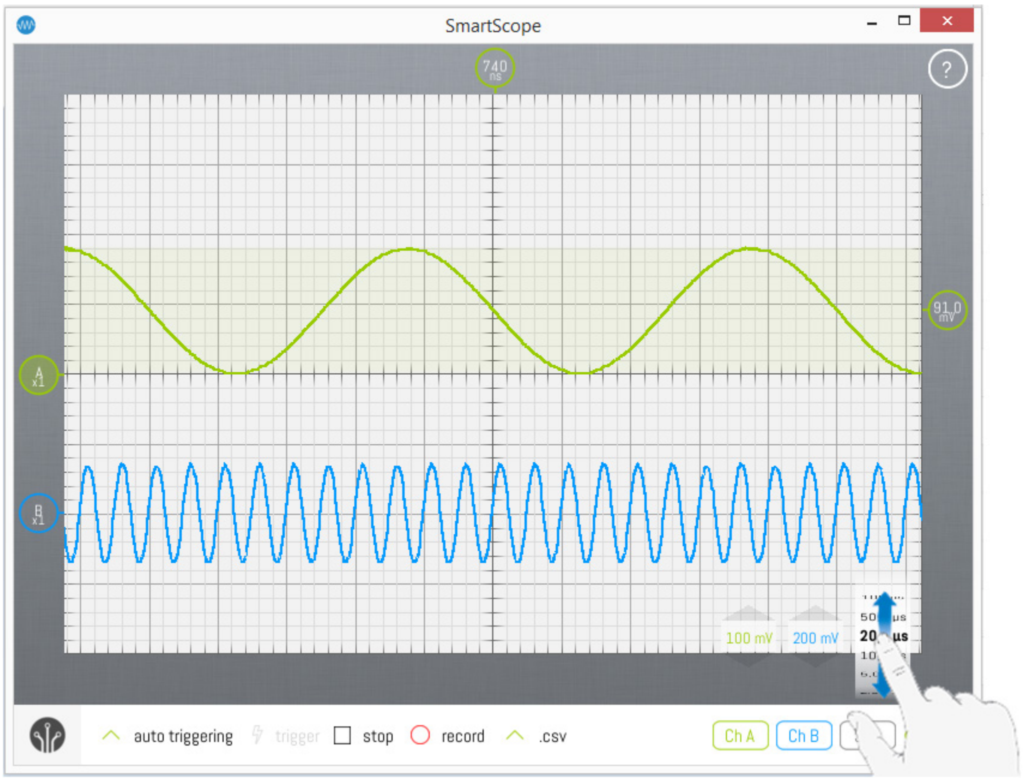
The grid can help you determine the timebase of your signal as well. The grid is split in 10 horizontal divisions, and the timespan of each division is displayed in the Timebase indicators, on the bottom-right of the screen. Notice that each signal can have a different scaling factor, and such there is one indicator for each wave.
The image below shows how timebase zooming on the green wave can be done:



Channel B

Offset	-3.00 V	Vertical	1.00 V/div
Max	1.05 V	Min	-1.04 V
Mean	20.7 mV	Peak-to-peak	2.09 V
RMS	714 mV		

Additionally, the timebase can also be adjusted using the picking wheel at the bottom-right of the screen. To do so, simply drag it up or down as shown in the image below:



3.4 Panning

While zooming allows you to fit the signal on the screen, you'll also want to position it to your liking. Here are the operations to achieve this:

Touch	Mouse	Keyboard
Drag	Left-drag	Arrow keys

Using touch/mouse, you can drag either the wave (notice the faintly colored background behind the wave: this indicates the sensitive area!), its 0-level indicator or the horizontal trigger indicator. Dragging the indicator has the benefit that they auto-snap to the grid divisions.
Dragging will also occur when, while pinching, you move both fingers in the same direction.

3.5 0-panning

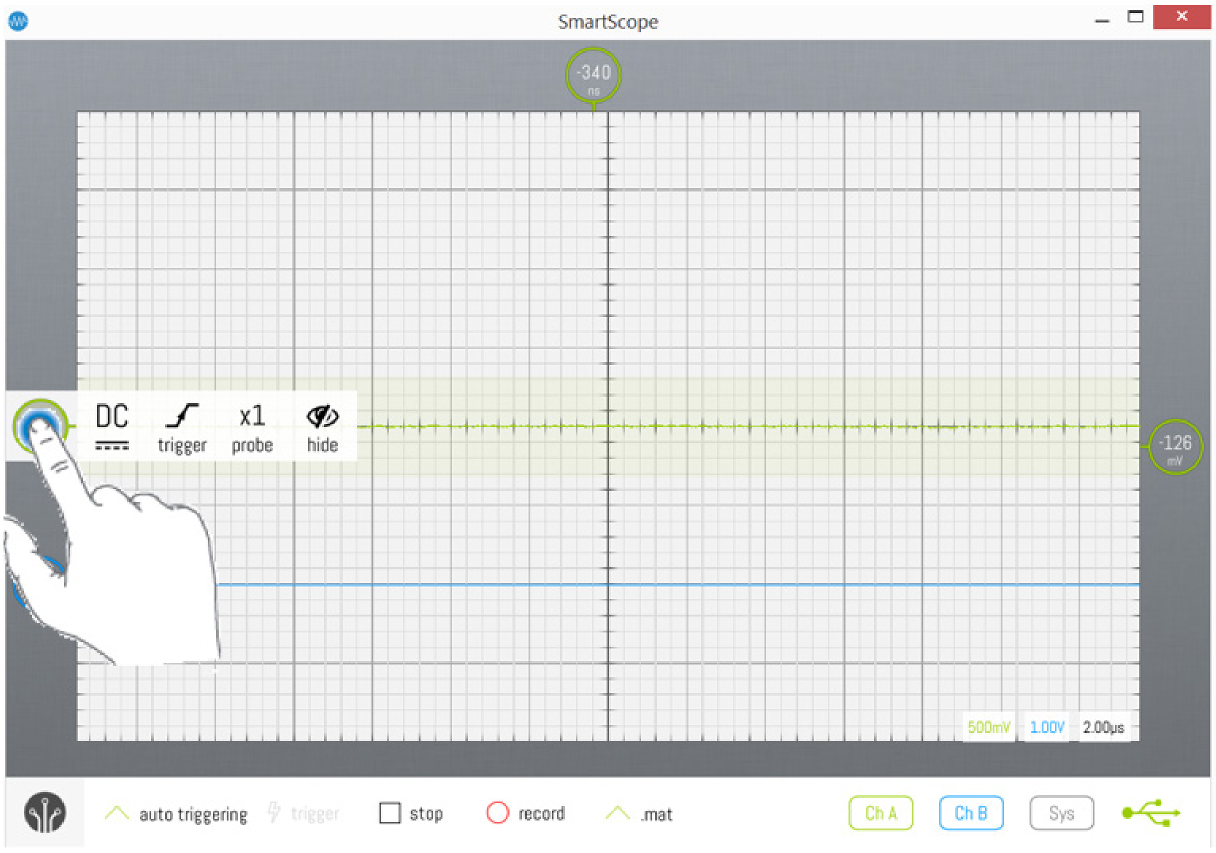
When using touch, you'll often want to set the timebase or voltage offset to 0. This can be done accurately in these ways:

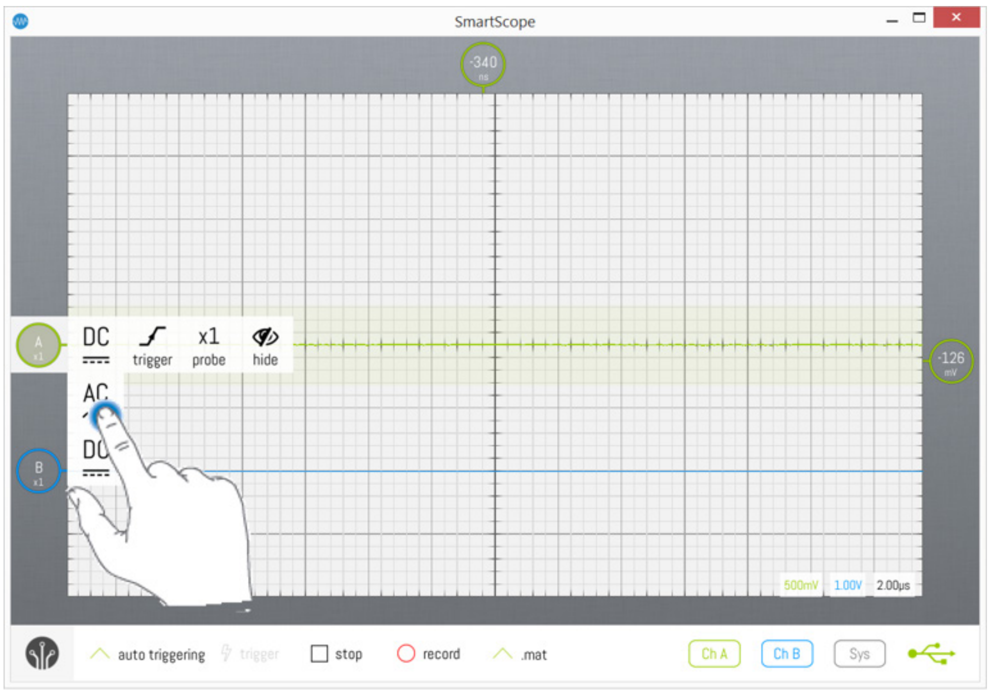
- By double-tapping the 0-level indicator, or double-tapping the horizontal trigger indicator
- By dragging the indicators, they will auto-snap to all grid divisions, including the 0-levels

4 AC and DC coupling

By default, each analog channel is set to DC coupling. This means that the voltages displayed will be the true, absolute voltages. Eg: a wave between 0V and 5V will be displayed as a wave between 0V and 5V. Alternatively, you can choose the set a channel to AC coupling. Doing so will cause the wave to be displayed, centered around 0V. Eg: the same wave between 0V and 5V will be displayed between -2.5V and 2.5V. This mode is quite often useful, eg in case you want to visualize a transient signal of low amplitude superposed on a large voltage. Eg: a 0.1V noise wave on top of a 12V power supply.

You can switch between DC and AC coupling, simply by clicking/tapping the wave indicator, and selecting the desired coupling mode. This is shown in the images below:





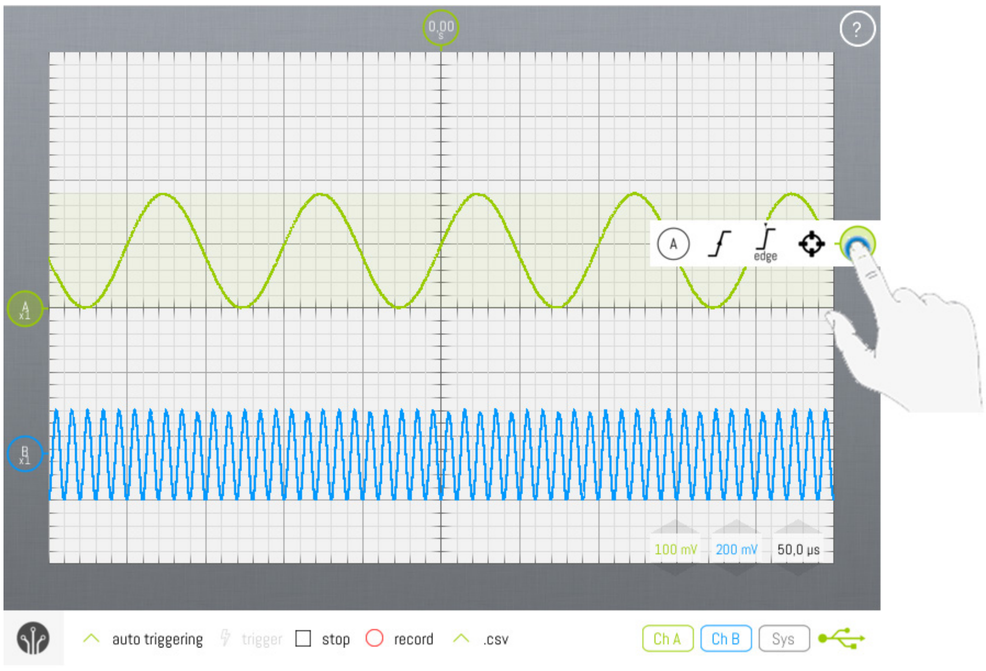
5 Triggering

Triggering is essential for two reasons:

- In case of repetitive signals: triggering will align all captured waveforms before displaying them, by positioning the crossoverpoint of captured waves at the same location on the screen
- In case of sporadic signals: the SmartScope will wait until an edge was detected, and then display that capture. Otherwise, the SmartScope would be capturing patches of data at random, and there would be a chance you would miss the section of interest.

The SmartScope supports simple edge-based triggering (rising/falling/any edge), but also more advanced triggering methods, see Advanced triggering options. This section will focus on basic triggering.

A rising edge is detected whenever the input signal transitions from below the trigger voltage to above the trigger voltage. This trigger voltage can be set as well (see Changing the Trigger Voltage, below).
All trigger settings can be set through the Trigger context menu, invoked by tapping on the Trigger voltage indicator or Trigger timebase indicator, as shown below:

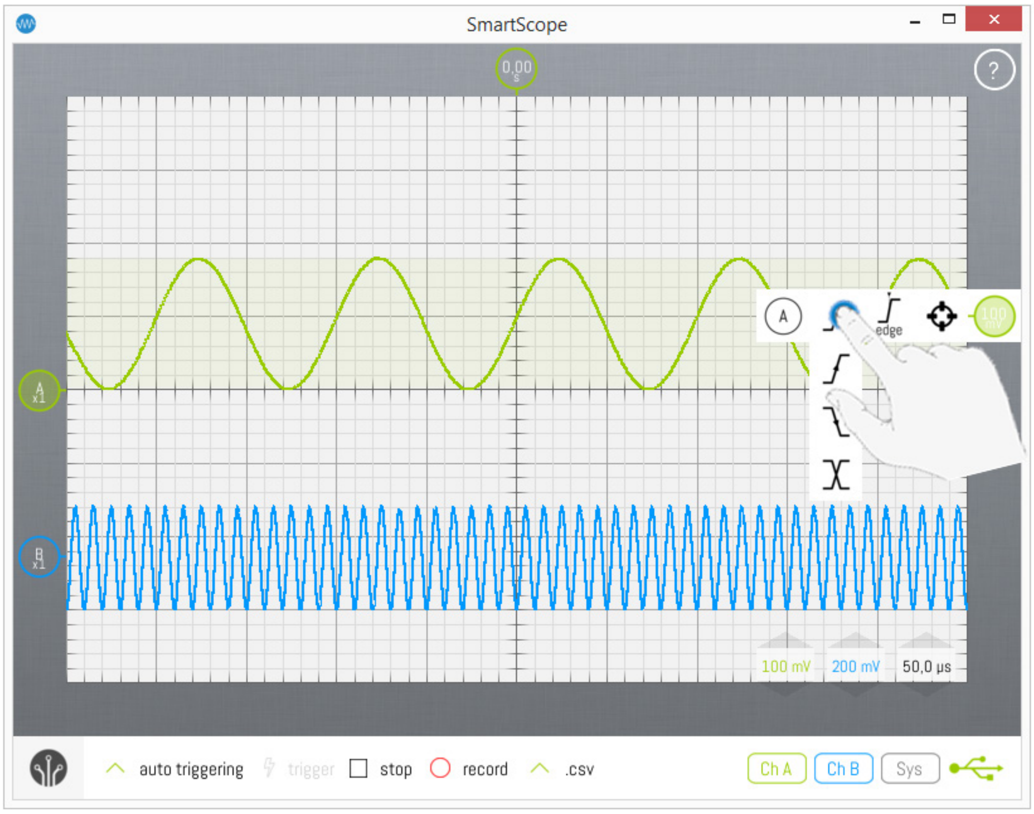


5.1 Changing trigger channel

You can specify which channel to trigger on, by opening the Trigger context menu as shown above, and tapping on the leftmost item. This will display a list of channels on which you can trigger. Changing trigger channel can also be done by hitting the T key when a keyboard is available.

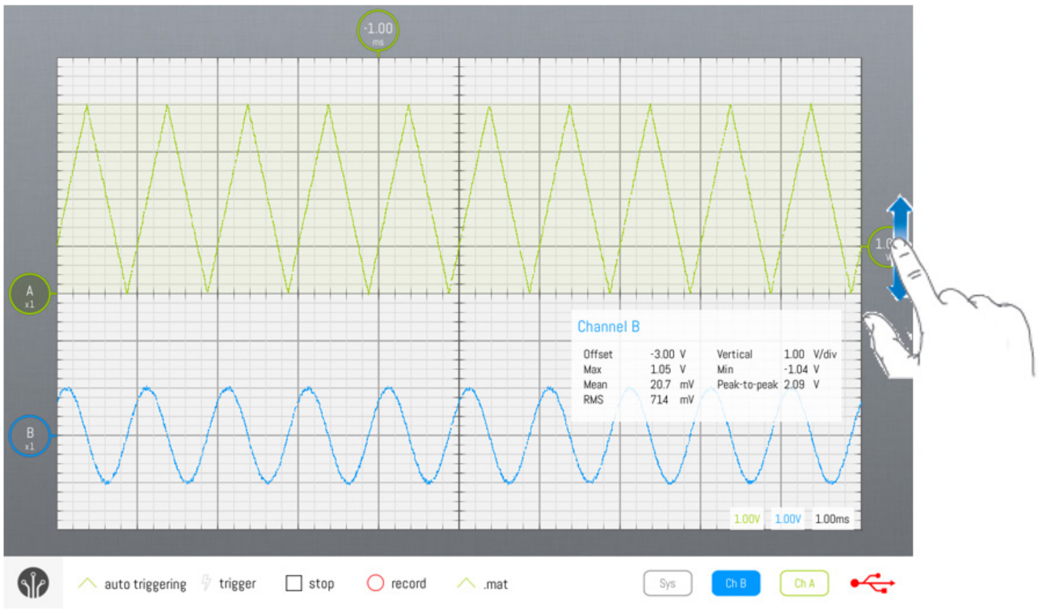
5.2 Changing between rising/falling/any edge

In the Trigger context menu, make sure the main trigger mode is set to Edge (shown in the image above). Next, you can specify which type of edge to trigger on, by opening the Trigger context menu as shown above, and tapping on the second leftmost item. A small drop-down menu will be shown, allowing you to specify whether on which edge you want to trigger. This is shown in the image below.



5.3 Changing the trigger voltage

The trigger voltage level can be changed intuitively by scrolling the Trigger level indicator on the right of the screen up or down, as shown in the image below. Notice the effect of triggering: the signal will cross the trigger level at the intersection of the Trigger voltage indicator and the Trigger timebase indicator.

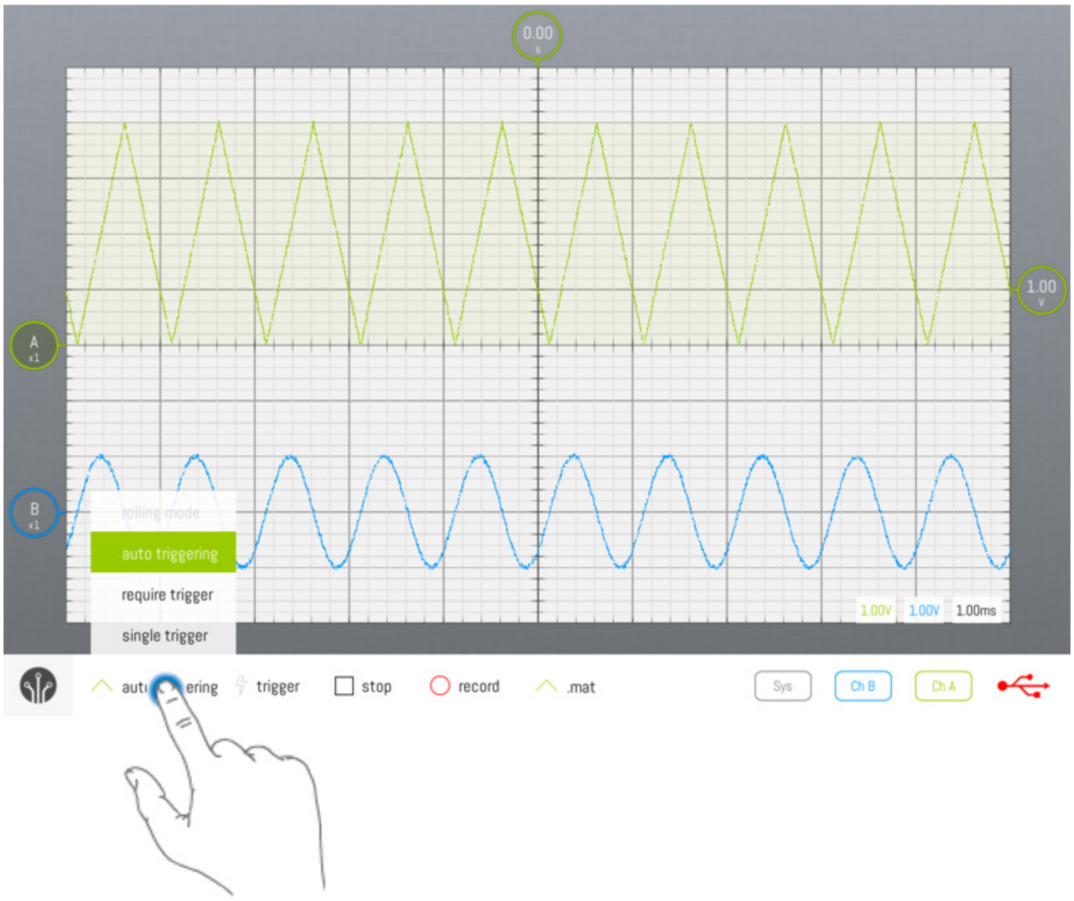


Channel B

Offset	-3.00 V		Vertical	1.00	V/div
Max	1.05 V		Min	-1.04	V
Mean	20.7 mV		Peak-to-peak	2.09	V
RMS	714 mV				

5.4 Changing the type of triggering

By default, the SmartScope will start using Auto triggering mode, and will switch to Rolling mode whenever you zoom the timebase beyond 20ms/div, to visualize slow signals. However, you can freely select the trigger mode of your choice by tapping on the Trigger type drop-up list, as shown below. Notice that the 'Rolling mode' is only available when you're visualizing slow signals (20ms/div and up).



5.4.1 Auto mode

In Auto mode, every time a trigger is detected, the waveform is acquired, transferred to the host and visualized. Whenever the trigger is lost (eg: when the input signal is removed, or when the signal changes so no more crossings with the trigger level occur), the SmartScope sends any acquired waveform to the host for visualization

5.4.2 Normal mode

In Normal mode, every time a trigger is detected, the waveform is acquired, transferred to the host and visualized, much like Auto mode. However, in case no trigger is detected, no new data is transferred nor visualized, keeping the last triggered waveform on the screen.

5.4.3 Single mode

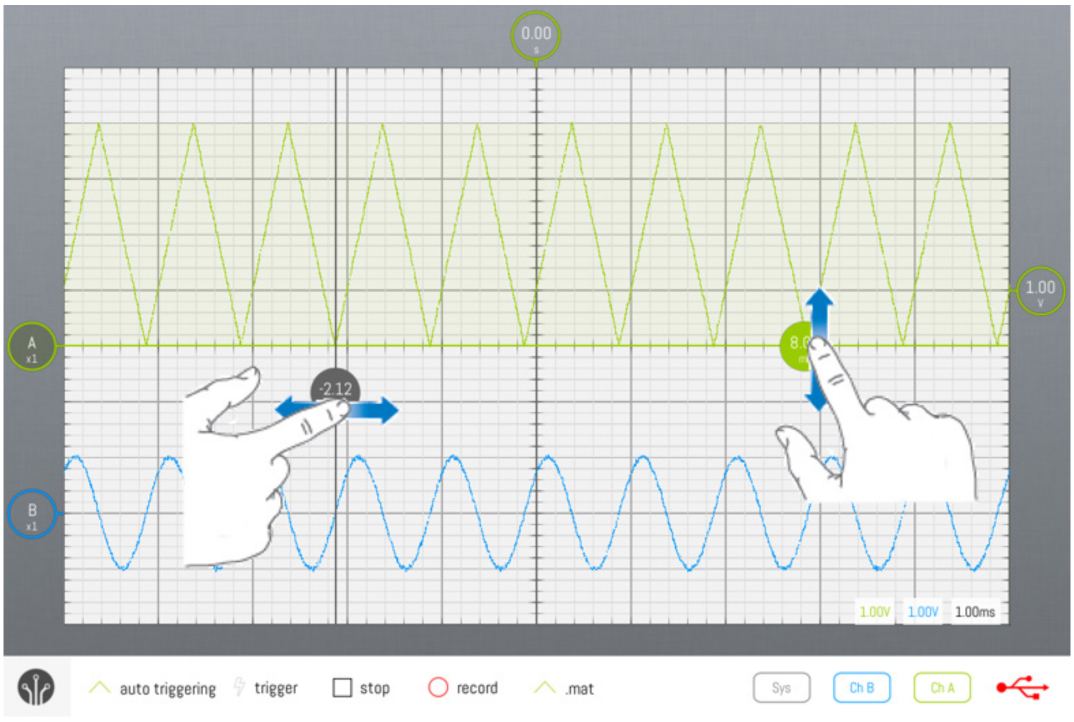
Single mode puts the SmartScope in waiting mode until a trigger is detected. In such case, the waveform is acquired and visualized, after which the SmartScope puts itself in Stopped mode. Single mode is useful in case you want to visualize the very first occurrence of a trigger, or in case you want to capture a unique event and don't want to remove the result in case of some glitch occurs when you remove the probe.

5.4.4 Rolling mode

When you're visualizing slow signals, you end up with timebase settings like 100ms/div. With 10 divisions, this means it takes 1 seconds before an acquitisition can be made, so you only get screen refreshes every second. In such case, it is usually preferred to visualize the data as a continuously incoming stream of data, as you get realtime updates of the signal and you can stop the SmartScope whenever you want.

6 Cursors

See the main page on Cursors to find out all details about them.

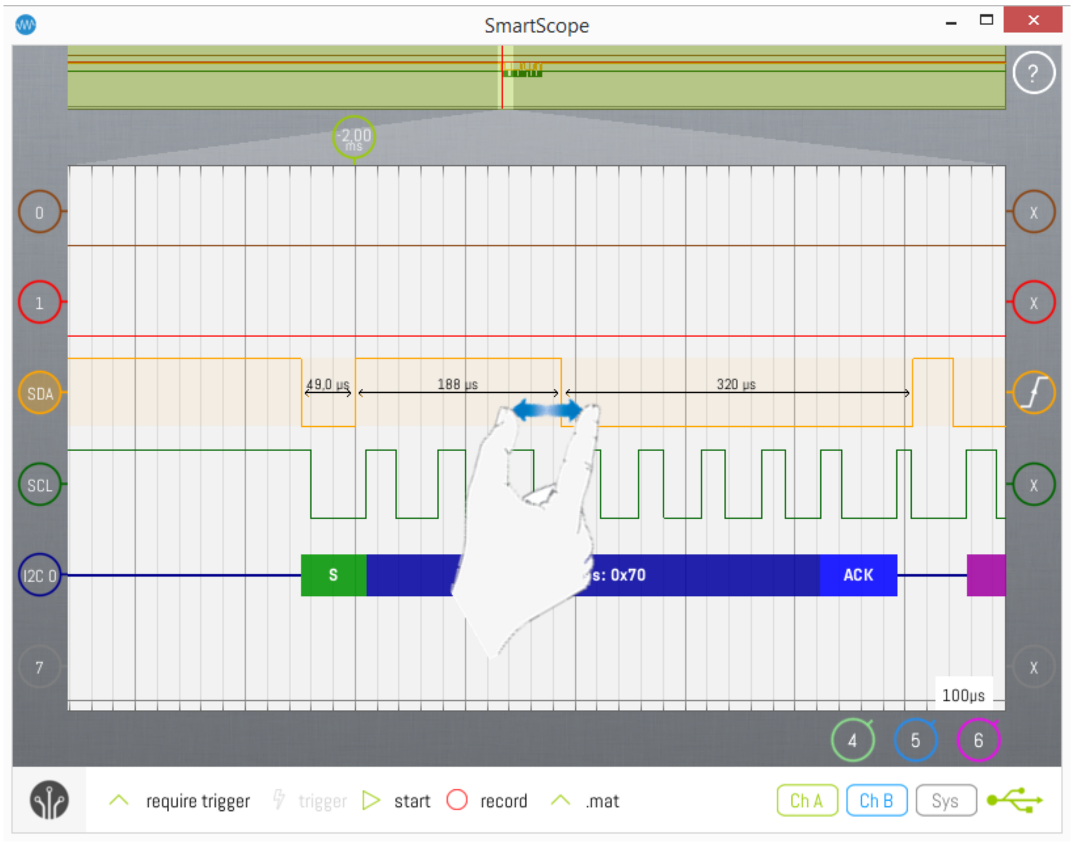


7 Measurements

For now, the SmartScope software contains a basic set of measurements. This number of measurements will grow, and together with it, the way of dealing with measurements will change. Measurements can relate to a channel (eg amplitude or mean voltage) or to the system (eg update rate). Measurement boxes can be shown or hidden by tapping on their corresponding button in the bottom-right corner. They can also be removed by dragging them out of the grid area.

Logic Analyzer functionality

Next to its dual-channel oscilloscope front-end, the SmartScope includes a powerful Logic Analyzer. 8 input pins on the back of the SmartScope allow you to sample a digital signals simultaneously at 100MS/s.
This section describes how to operate the SmartScope software to effectively use the Logic Analyzer functionality.



Contents

1 Logic Analyzer - Hardware specifications

- Supported digital voltage levels:
    - 3.3V, 5.0V: Up to 50MHz digital signals
    - 2.5V: Up to 40Mhz digital signals
    - 1.8V: Up to 10Mhz digital signals
- Maximum voltage range: [-0.5V, 5.5V]
    - Applying voltages beyond this range can permanently damage the input channel!

- Spikes/Transients outside this range are OK, as we have hi-speed bidirectional TVS diodes on each input

See the Connectors pinout article to find out which pins are reserved for the Logic Analyzer.

2 Selecting the Logic Analyzer mode

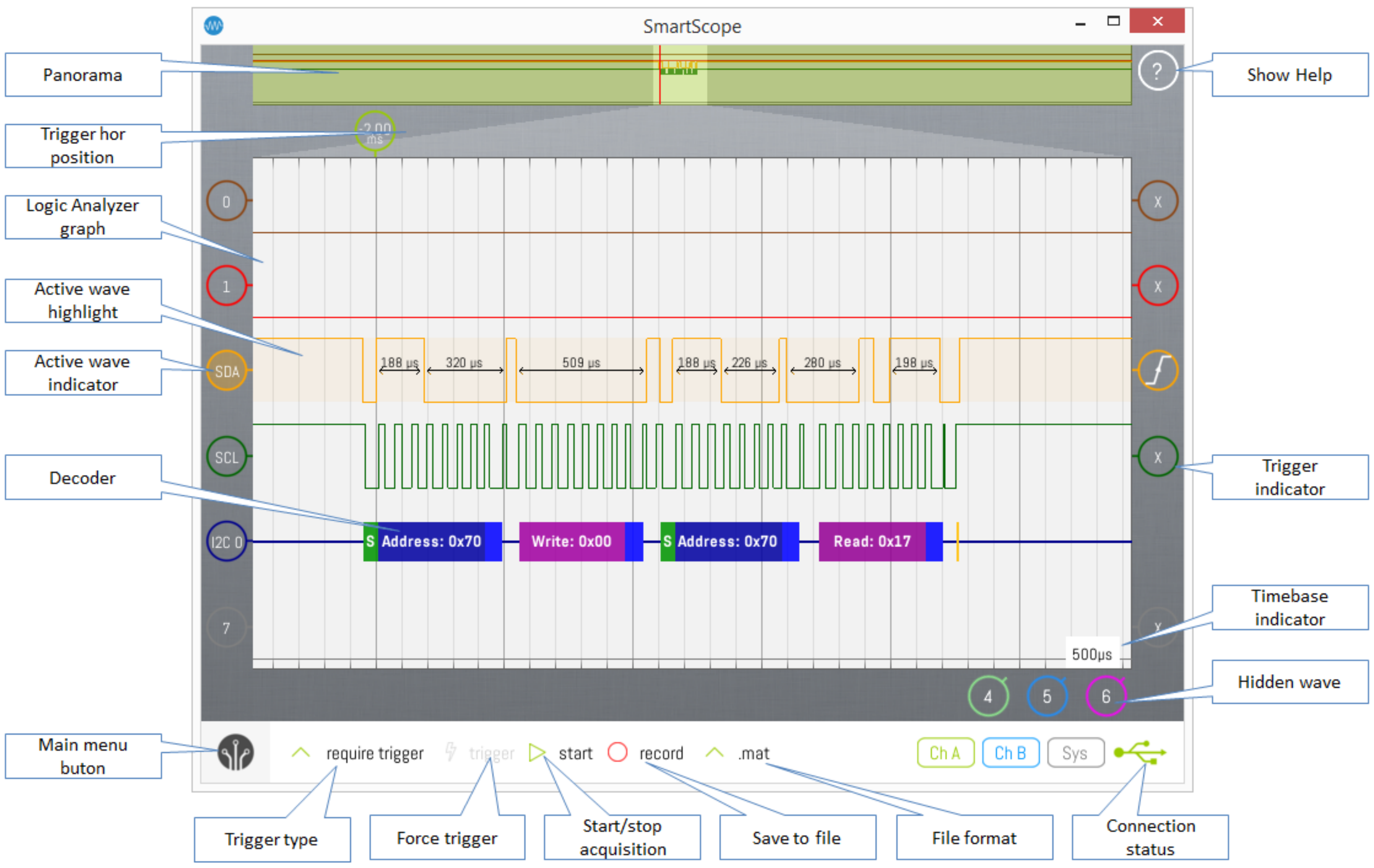
Simply open up the main menu, and select ''Digital mode'' to bring up the Logic Analyzer Graph. Note that the full Analog Graph functionality is also available in ''Mixed mode''
File:SelectLA.png

3 Logic Analyzer main screen

The key parts of the Logic Analyzer screen are shown on the image below. This image can be used as reference for the rest of this text.

Some elements are the topic of other articles:

- Panorama (RAM zoom) functionality
- Using the Protocol Decoders



4 Visualizing your digital signals

First you'll want to make sure you have optimal visibility on your digital wave. Since you're probably using the LA for visualizing digital communication, you'll want to

- Move the trigger position (pan) so the start of the communication is shown on the graph
- Change the timescale so the key part of the communication is fitting inside the graph

These actions are described below in more detail.

4.1 Changing the trigger position (panning)

You'll also want to position your waves to your liking. Here are the operations to achieve this:

Using touch/mouse, you can drag your waves or the horizontal trigger indicator. Dragging the indicator has the benefit that it auto-snaps to the grid divisions.

Touch	Mouse	Keyboard
Drag	Left-drag	Arrow keys

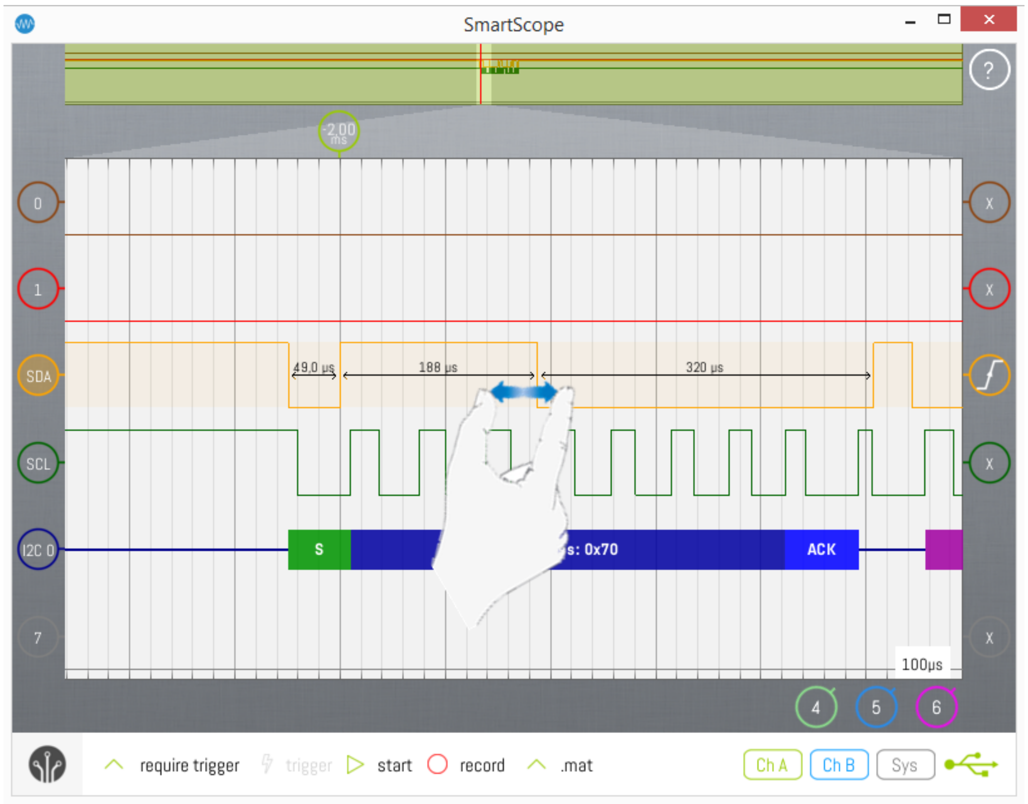
Dragging will also occur when, while pinching, you move both fingers in the same direction.

4.2 Timebase scaling

Depending on whether you want to visualize a fast signal or a slow signal, you'll want to set the timebase accordingly. This can be achieved using the following operations:

Touch	Mouse	Keyboard
Horizontal Pinch	Mousewheel or right-drag	Home or End

Zooming by mouse or keyboard will switch between predefined values. By pinching you can continuously zoom in or out to an arbitrary value. The image below shows how timebase zooming on can be done:



The grid can help you determine the timebase of your signal as well. The grid is split in 10 horizontal divisions, and the timespan of each division is displayed in the Timebase indicators, on the bottom-right of the screen.

5 Repositioning/Hiding/Showing signals

In some cases where you don't need 8 digital inputs, you might want to reposition or even hide some of them as they can clutter the screen. Repositioning is pretty straightforward: simply drag any digital wave or its indicator to wherever you desire. When you release the wave, all waves will auto-snap to their optimal position. You can also hide inactive digital waves, which will free up graph space for other signals. This is especially useful in Mixed mode, where the graph area can be small. When you hide a digital wave, the remaining signals will be enlarged. To hide a digital wave, tap its indicator as shown below, and select hide. The wave will be hidden, and its indicator will be moved to the bottom-right corner. To re-activate a hidden wave, simply tap its indicator residing at the bottom-right corner.

5.1 Time interval arrows / Active wave

The Logic Analyzer contains a feature which indicates the time between various rising or falling edges. This is only shown for the currently selected Active wave (see below) and for the wave underneath the mouse pointer. In order to select another wave as Active wave, simply tap/click that wave. Use (shift+)Tab to Activate the (previous)next wave. You can recognize the currently selected Active wave as follows:

- Its background is slightly highlighted
- Its 0-level indicator has a darker color
- Time interval arrows are shown between its rising and falling edges

Time interval arrows are only shown in case there is enough space. As an example, no arrows will be shown for a high-frequency clock when you're not sufficiently zoomed in on the time-axis.

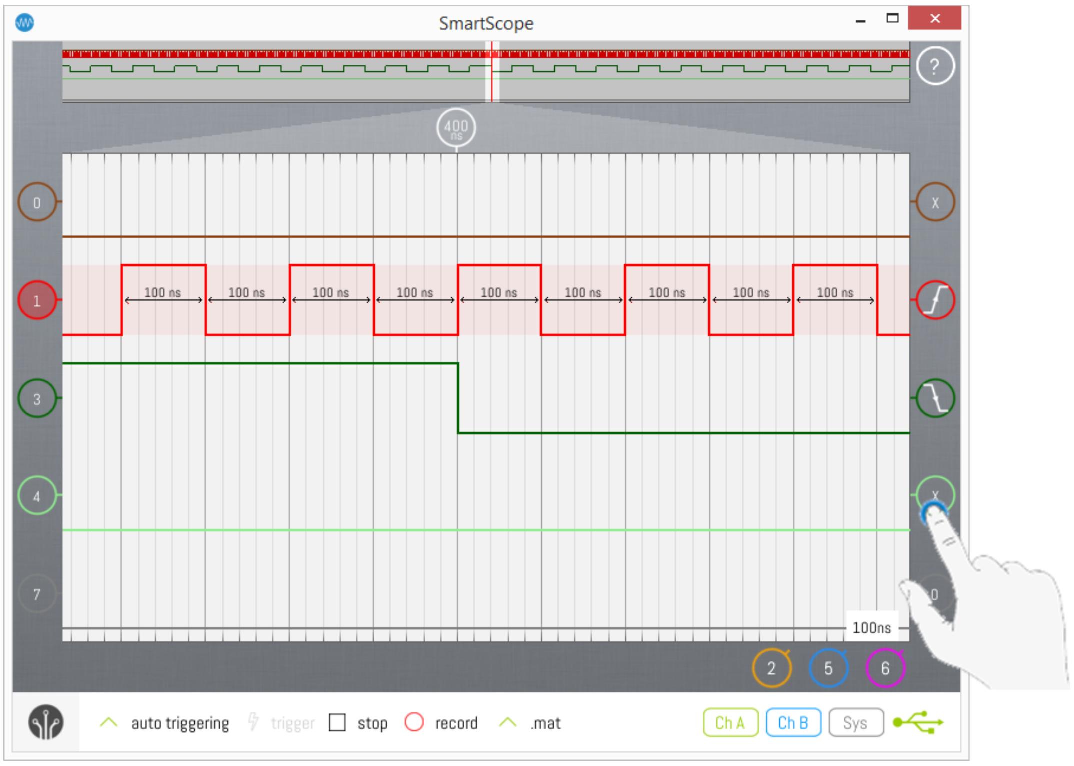
6 Triggering

The Logic Analyzer allows you to trigger on any combination of states/events of the 8 digital inputs. These include:

- Don't care (X)
- Rising (R)/falling (F) edge
- Low (0) / High (1)

For example, in the screen below the SmartScope is instructed to trigger at any event where:

- D1 encounters a Rising edge
- D3 encounters a Falling edge
- D7 stays Low



In order to change the trigger condition, simply tap the trigger indicators (also shown in the image above) until you reach the condition you require.

7 Trigger modes

The following trigger modes are available:

7.1 Auto mode

In Auto mode, every time a trigger is detected, the waveform is acquired, transferred to the host and visualized. Whenever the trigger is lost (eg: when the input signal is removed, or when the signal changes so no more crossings with the trigger level occur), the SmartScope sends any acquired waveform to the host for visualization

7.2 Normal mode (=required trigger)

In Normal mode, every time a trigger is detected, the waveform is acquired, transferred to the host and visualized, much like Auto mode. However, in case no trigger is detected, no new data is transferred nor visualized, keeping the last triggered waveform on the screen.

7.3 Single mode

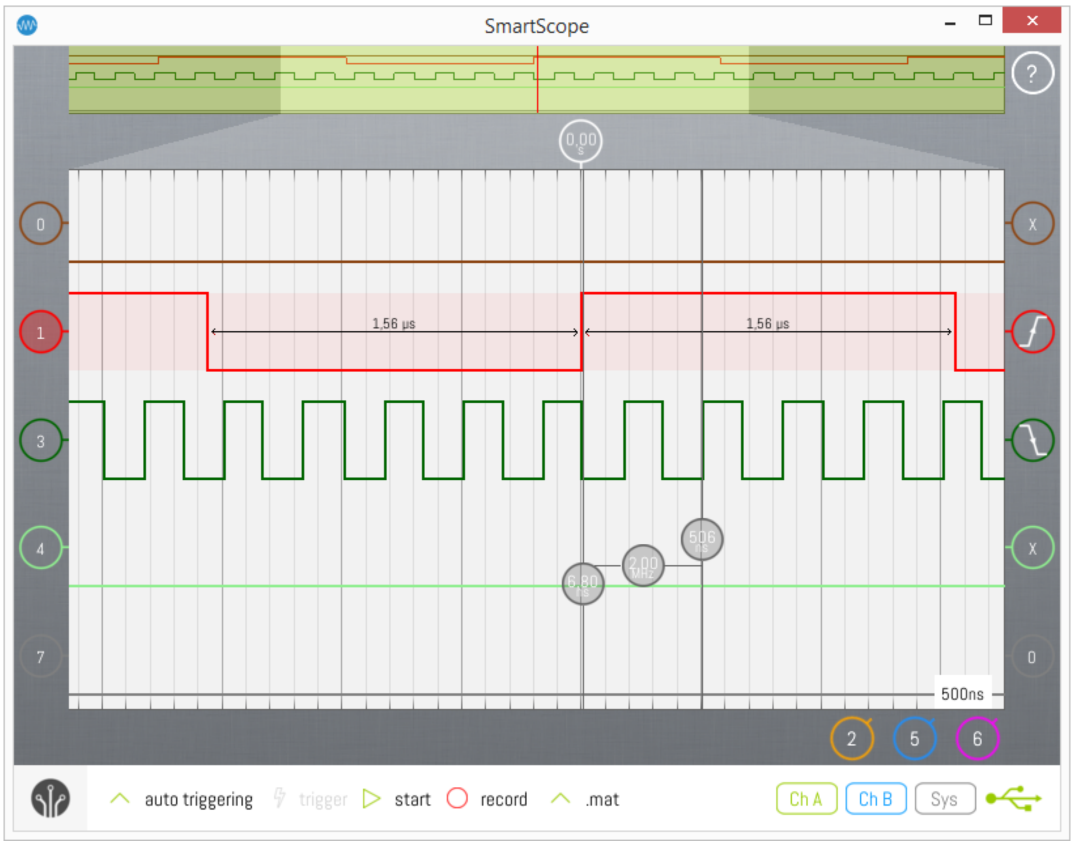
Single mode puts the SmartScope in waiting mode until a trigger is detected. In such case, the waveform is acquired and visualized, after which the SmartScope puts itself in Stopped mode. Single mode is useful in case you want to visualize the very first occurrence of a trigger, or in case you want to capture a unique event and don't want to remove the result in case of some glitch occurs when you remove the probe.

7.4 Rolling mode

When you're visualizing slow signals, you end up with timebase settings like 100ms/div. With 10 divisions, this means it takes 1 seconds before an acquitisition can be made, so you only get screen refreshes every second. In such case, it is usually preferred to visualize the data as a continuously incoming stream of data, as you get realtime updates of the signal and you can stop the SmartScope whenever you want.

8 Cursors

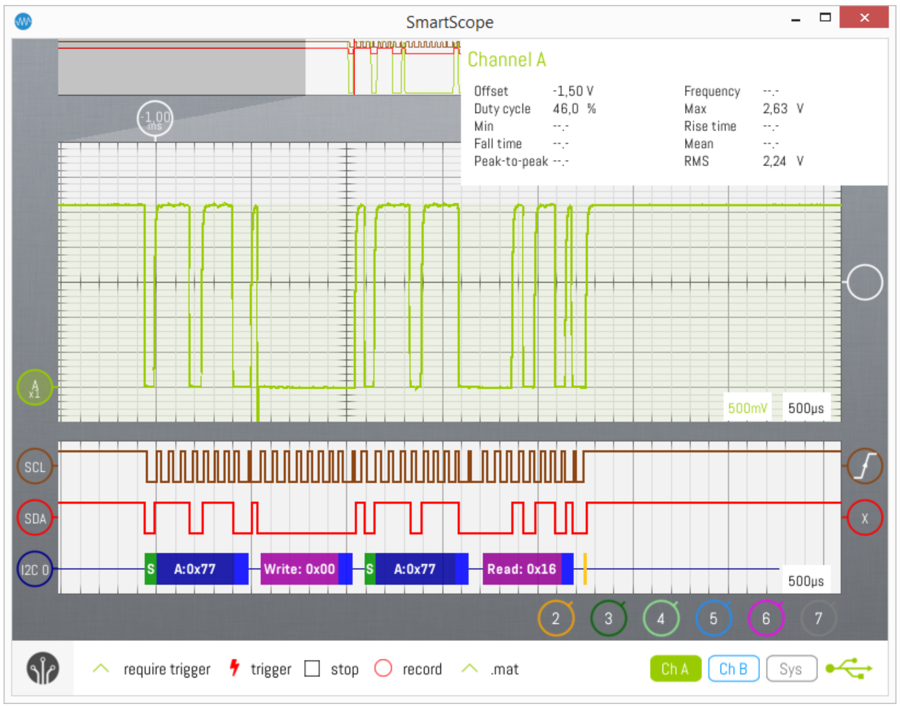
See the main page on Cursors to find out all details about them. Only vertical cursos are supported on the Logic Analyzer graph.



Mixed mode

In some cases, it can be very useful to use the Oscilloscope and Logic Analyzer simultaneously.

An example is shown below where the LA is showing a correct I2C communication, but your chips are not functioning properly.
Investigating the signal with the Oscilloscope, you can immediately see your I2C host is operating at 2.6V, while your I2C slave is expecting 3.3V.



Channel A

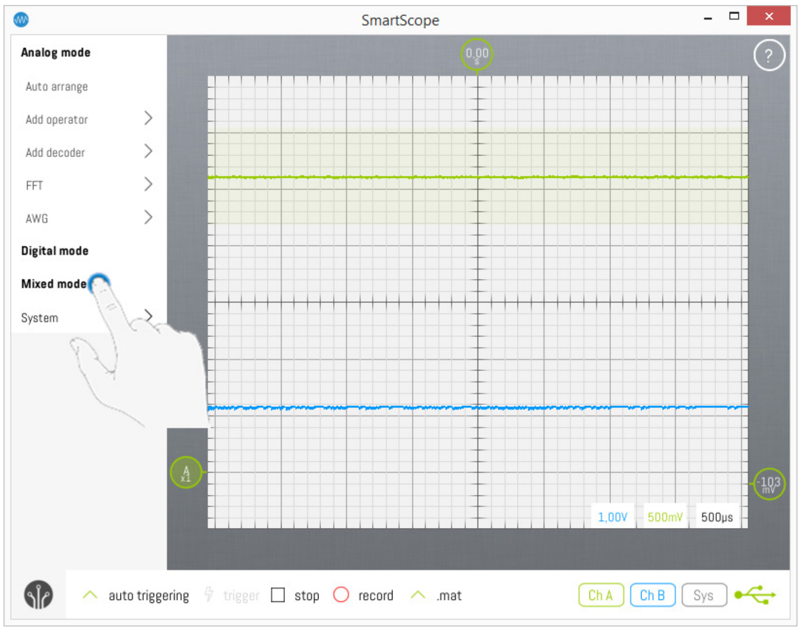
```
Offset        -1,50 V        Frequency   --.-
Duty cycle    46,0  %        Max         2,63  V
Min           --.-           Rise time   --.-
Fall time     --.-           Mean        --.-
Peak-to-peak  --.-           RMS         2,24  V
```

S A:0x77 Write: 0x00 S A:0x77 Read: 0x16

Contents

1 Selecting Mixed mode

Simply open up the main menu, and select ''Mixed mode'' to bring up both the Analog Graph and the Logic Analyzer Graph.



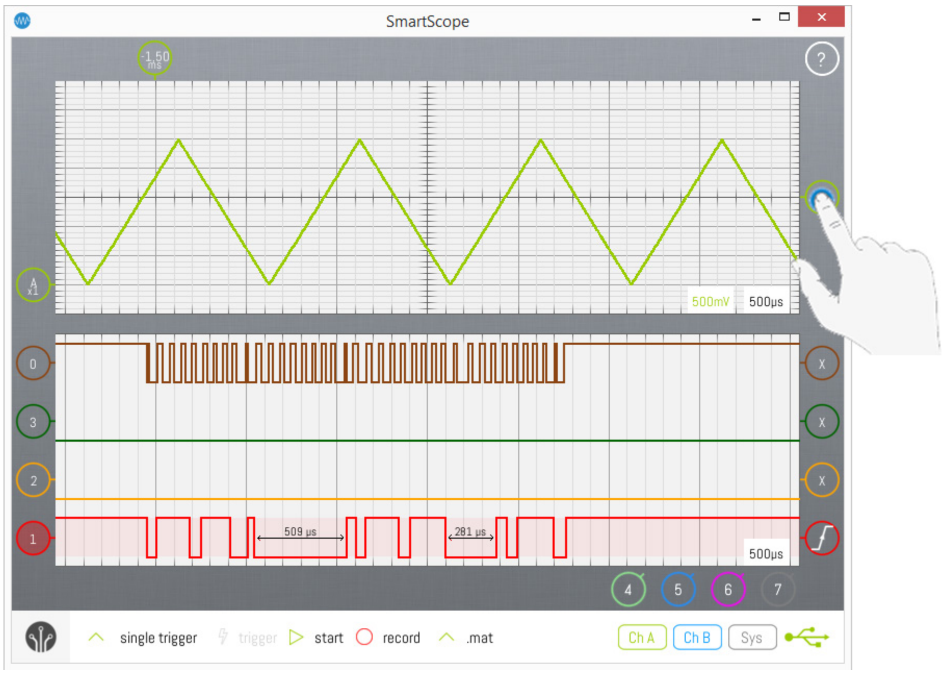
2 Operating the Analog and Logic Analyzer graphs

Both graphs function exactly the same as in pure Analog or Digital mode, which are explained in full detail on the following pages:

- Oscilloscope functionality
- Logic Analyzer functionality

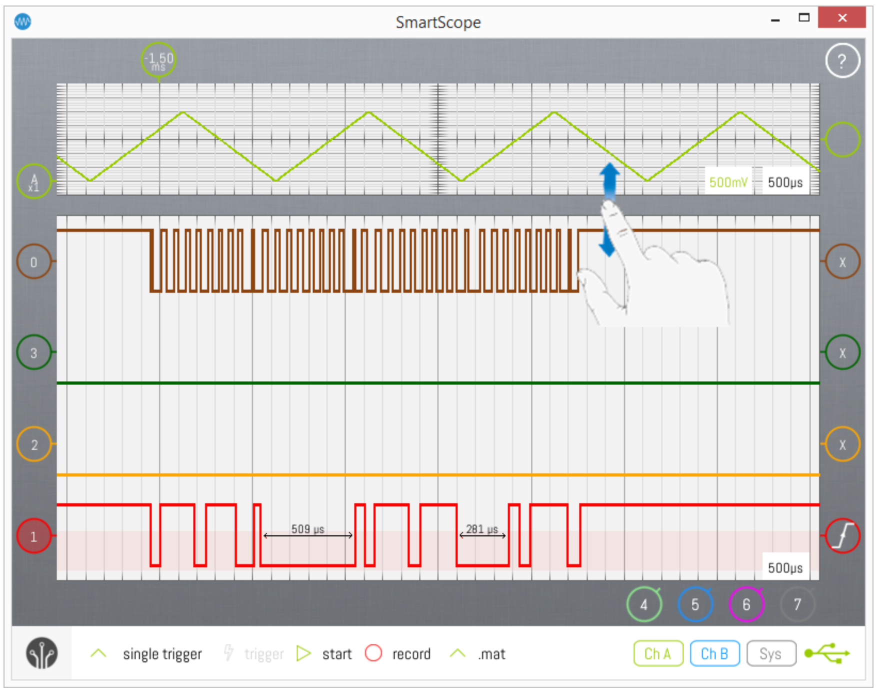
3 Changing between Analog/Digital Trigger mode

In Mixed mode, you use either the Analog or Digital trigger. In either triggering mode, the other trigger indicators are shown on the right of their graphs, but they are empty. Simply tap any trigger indicator to activate that trigger mode. As an example, Digital trigger mode is activated in the image below. In order to activate Analog trigger mode, simply tap the empty analog trigger indicator to the right of the Analog graph.



4 Changing size of graphs

Since screen space can be limited on phones or tablets, you'll often want to enlarge one graph and shrink the other. This can be done by dragging the area between both graphs, as shown in the image below:



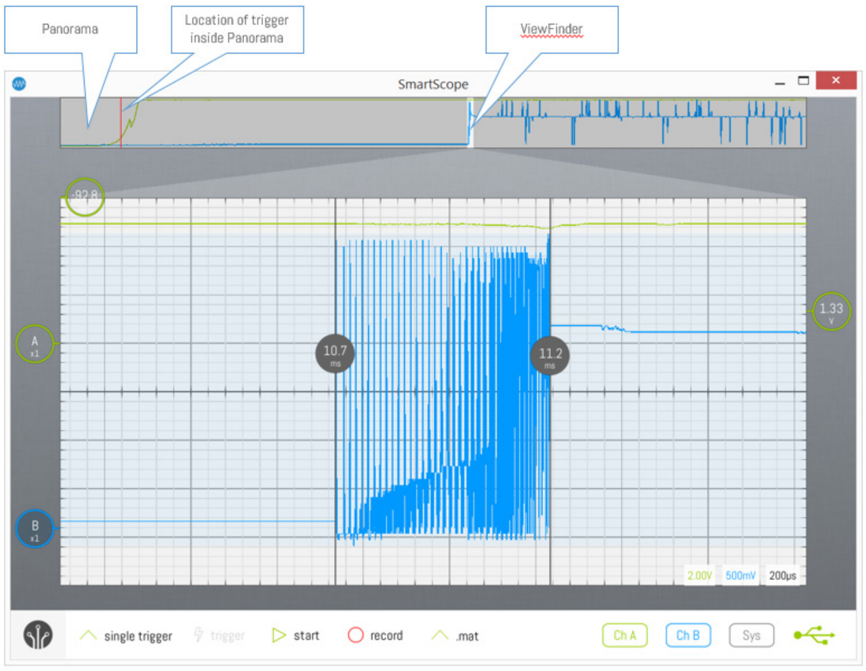
Panorama (RAM zoom) functionality (available on SW version v0.0.8 and above)

The SmartScope comes with 64Mbit of RAM, which by default is used to provide a 4M (4.195.304) sample buffer for each channel. In general, this means that whenever you stop an acquisition, you can zoom in about 1000 times on the waves currently displayed. This can be very useful in case you want to trigger on an event, and study in fine detail all causing and resulting effects which happened before or after that event.

Contents

1User Graphical Interface elements

An example screenshot is presented below, after which the main elements will be described below.



The Panorama

The Panorama is the bar shown at the top of the screen. It displays the entire contents of the RAM memory.

The ViewFinder

The ViewFinder is the area of the Panorama corresponding to the area currently displayed on the main screen. It is highlighted on the Panorama, to give a clear **visual indication** of which section of the Panorama you're currently inspecting on the main screen. This is further enhanced by the highlights underneath the Panorama, visually linking the area of the main graph to the ViewFinder.

The ViewFinder can be moved across the Panorama by swiping it, dragging it by mouse, or by CTRL+Left/Right.

The Trigger

The position of the trigger is also shown in the Panorama. In case the trigger is outside the timespan of the main graph, this gives a quick indication of where the trigger is relative to the signal visualized in the main graph.

2 Using the Panorama and ViewFinder

Using Touch and Mouse, you can directly manipulate the element of the visual interface. Using Keyboard, all operations acting directly on the Panorama and ViewFinder are accessed by holding the CTRL key pressed.

The following table can be used as quick reference:

	Touch	Mouse	Keyboard
Regular Zoom in (without changing ViewFinder size)	Pinch main graph	Mousewheel scroll on main graph	End
Regular Zoom out (without changing ViewFinder size)	Pinch main graph	Mousewheel scroll on main graph	Home
ViewFinder Zoom in (increases ViewFinder size)	Pinch Viewfinder	Mousewheel scroll on ViewFinder	CTRL+End
ViewFinder Zoom out (decrceases ViewFinder size)	Pinch Viewfinder	Mousewheel scroll on ViewFinder	CTRL+Home
Move ViewFinder	Drag ViewFinder	Drag ViewFinder	Ctrl+Left/Right
Show the Panorama	Double-tap top border	Double-click top border	P
Hide the Panorama	Double-tap Panorama	Double-click Panorama	P

3 Help! I'm getting terribly slow refresh rates

In certain configurations, using the Panorama might result in undesirably slow refresh rates.

Let's apply some simple math on the screenshot at the top of this page. The main graph is set to 200us/div, resulting in 1,2ms shown on the total screen, so you could expect up to 800 updates per second. However, notice that the full Panorama is much larger than the currently set ViewFinder: about 1%. This means that it takes 120ms to fill a new Panorama with samples! As such, the maximum refresh rate of the screen will be 8Hz.

In case this is undesired, either:

- Enlarge the ViewFinder using CTRL+Home and afterwards Zoom in on the global timescale using 'Home'
- Hide the Panorama using 'P'

4 Under the hood

4.1 When is the Panorama shown/hidden?

In case the Panorama is hidden, and you stop the acquisition of the SmartScope, the Panorama is automatically shown, inviting you to browse through the data stored in the RAM on the SmartScope. When you continue the aquisition without changing anything to the ViewFinder, the Panorama is hidden again. Other than those cases, the Panorama will not hide or show itself automatically.

In case you don't use the Panorama and want to free up the screen area it is consuming, simply press the 'P' button or double-click on the Panorama. When hiding the Panorama, the buffersize is minimized to correspond to the main graph, making sure the maximum data refresh rate is achieved.

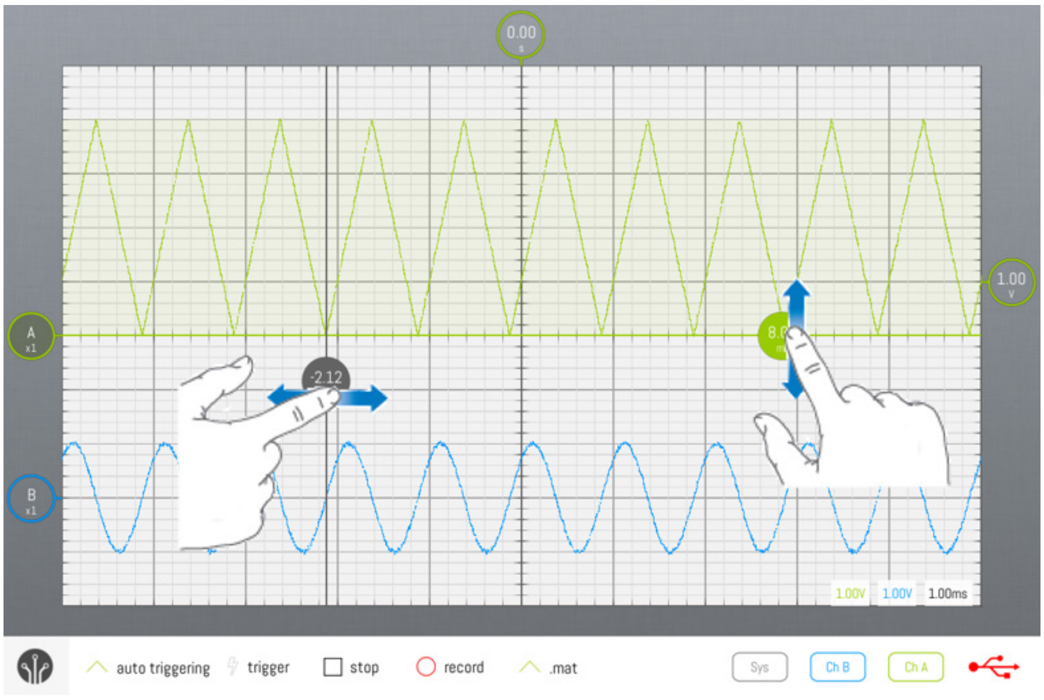
4.2 Acquisition speed and buffer size details

By opening up the System measurement box at the bottom-right of the screen, you can find 4 parameters controlled by the Panorama and ViewFinder configuration:

- Acquisition: shows the timespan of data stored inside the RAM
- VP Length: shows the timespan represented by the ViewFinder
- VP Offset: shows the time offset of the ViewFinder, relative to the first sample of the Panorama
- Sample rate: the rate at which analog voltages and/or digital signals are acquired and stored inside the RAM. In case the Panorama size is set larger than 4e6 samples, the sample rate will automatically be downscaled to the optimal speed.

Cursors

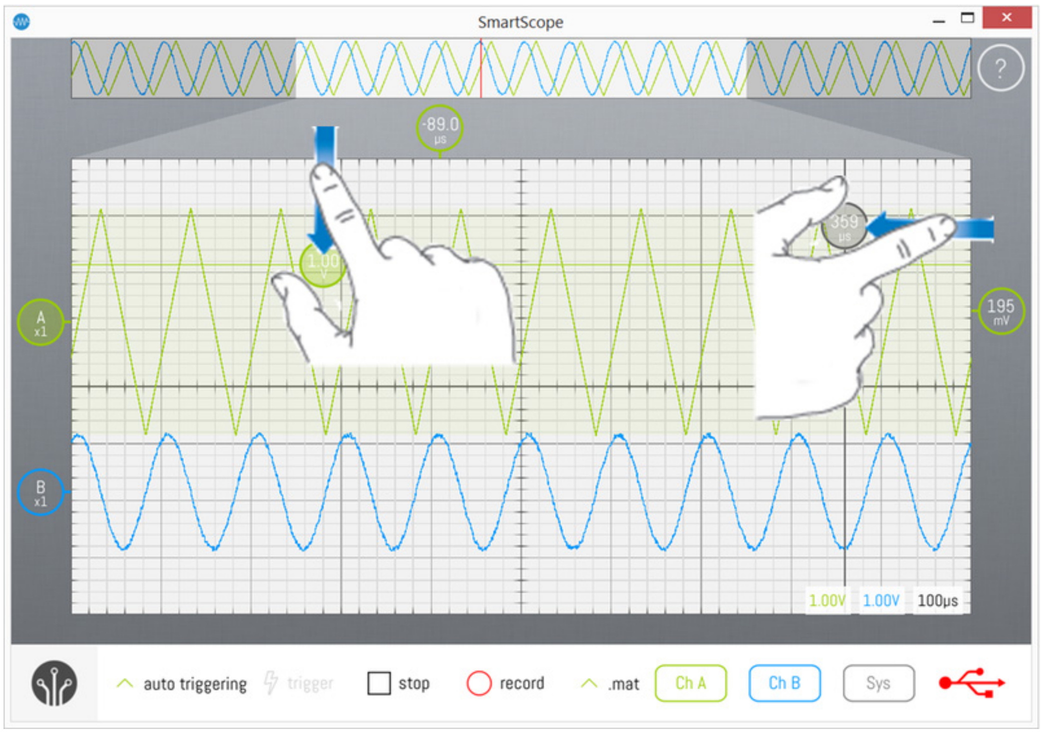
Cursors can be extremely handy for quickly finding a voltage or timestamp. Horizontal cursors show a voltage related to a waveform, and therefore have the color of the waveform they belong to.
Cursors are **available on both the Analog graph and the Logic graph**. Simply slide them in from the right.



Contents

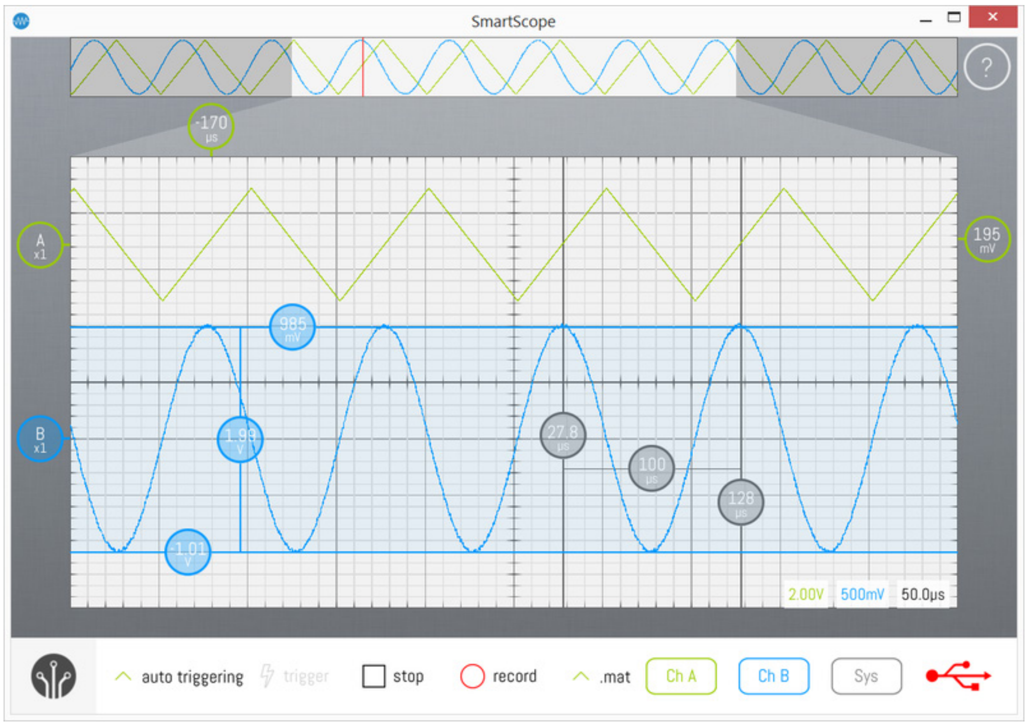
1 Adding and Removing cursors

Usually cursors are very well hidden inside the UI of oscilloscopes. In case of the SmartScope, simply start dragging from outside the grid area (on the gray border) towards the inside of the grid area, and a cursor will appear. To removed a cursor, simply drag it off the grid onto the gray border again.



2 Delta cursors

Whenever you drag 2 cursors of the same type onto the main graph, a delta cursor will appear showing the difference between its two reference cursors. This delta cursor can be moved freely around the main graph, as long as you stay between its 2 reference cursors. Like any other cursor, you can remove the Delta cursor by dragging it off the main graph onto the grey borders. To make it appear again, simply tap one of the reference cursors.

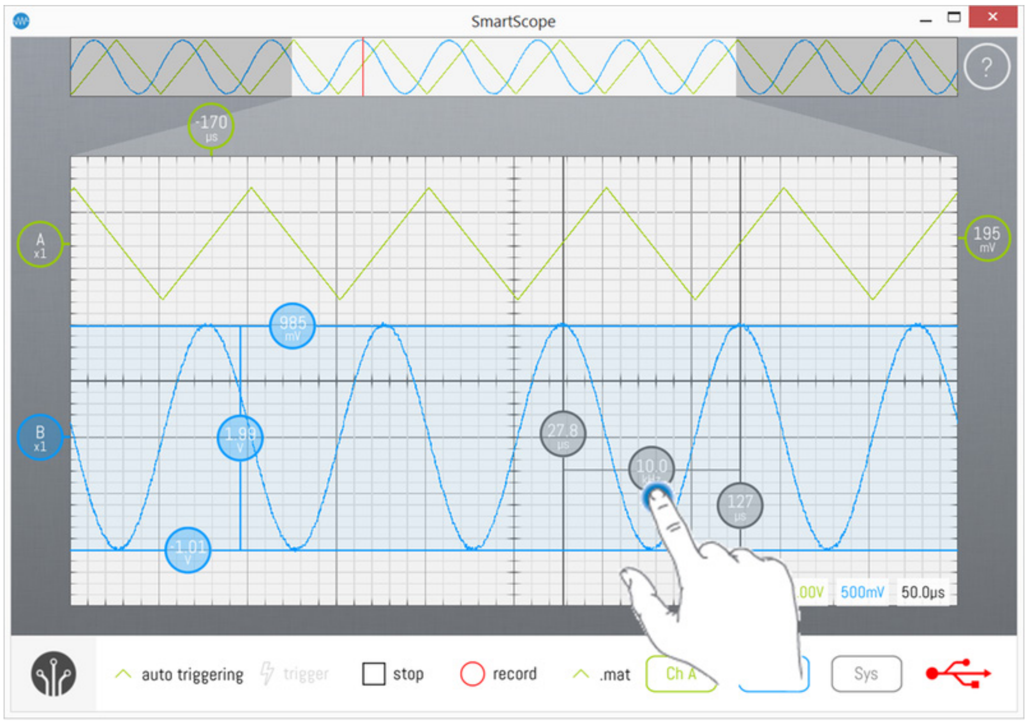


2.1 Changing Delta reference cursors

In case you have more than 2 cursors of the same type, the delta cursors will by default use the 2 most recently added cursors as references. If you want to use an earlier cursor as reference cursor, simply tap that cursor and the delta cursor will use it as reference.

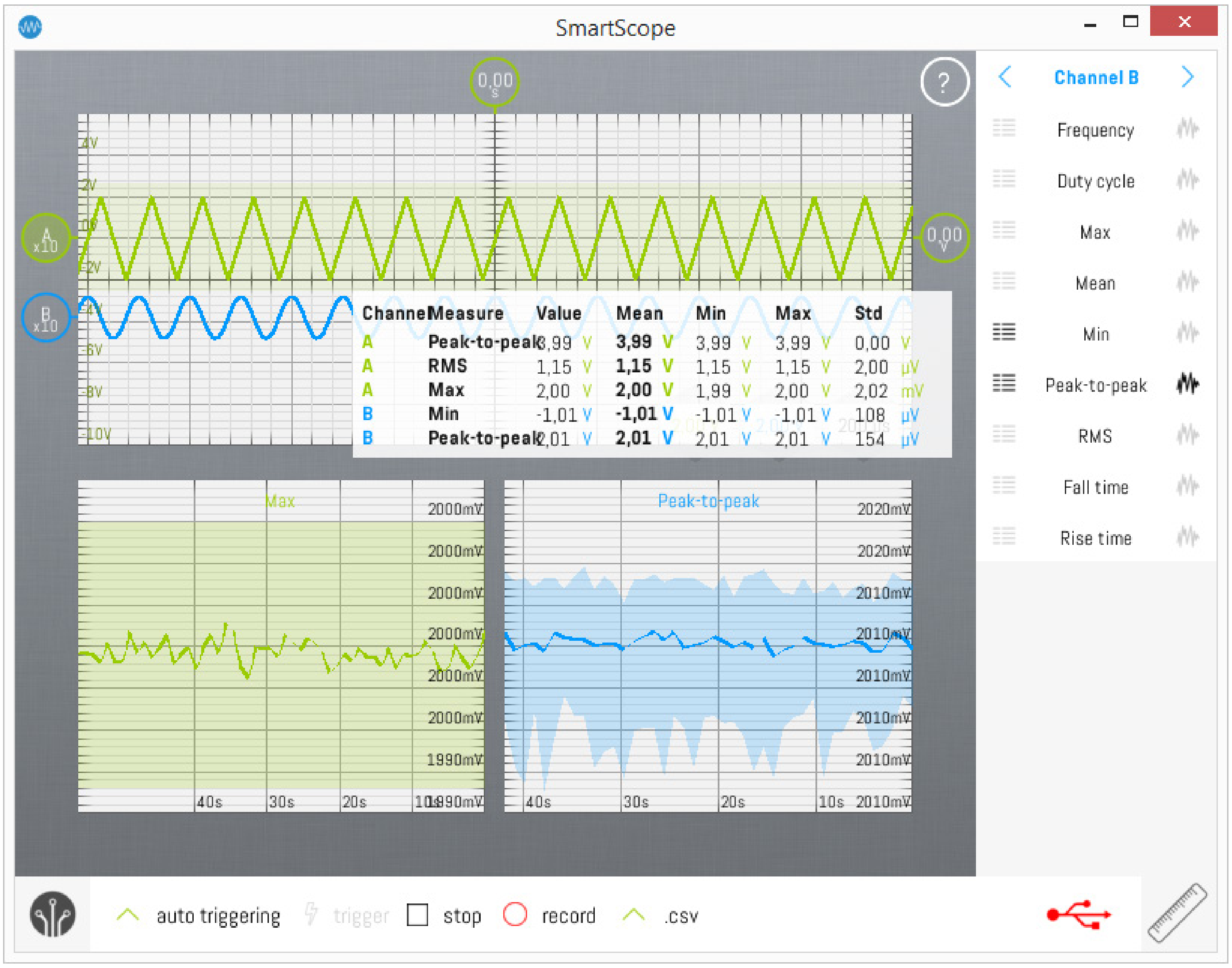
2.2 Toggling Delta cursor between Time and Frequency indication

Often, when checking the difference between 2 locations on the time-axis, you're more interested in the corresponding frequency rather than the actual time difference itself. Simply tap on the delta indicator to make it toggle between Time and Frequency indication.



Measurements

Since v0.13.0.0, the SmartScope includes a more advanced Measurement subsystem. Each measurement now shows more information (min/max/mean/std), and you can even watch how these values change over longer periods of time (even hours!) inside the SmartScope app itself.

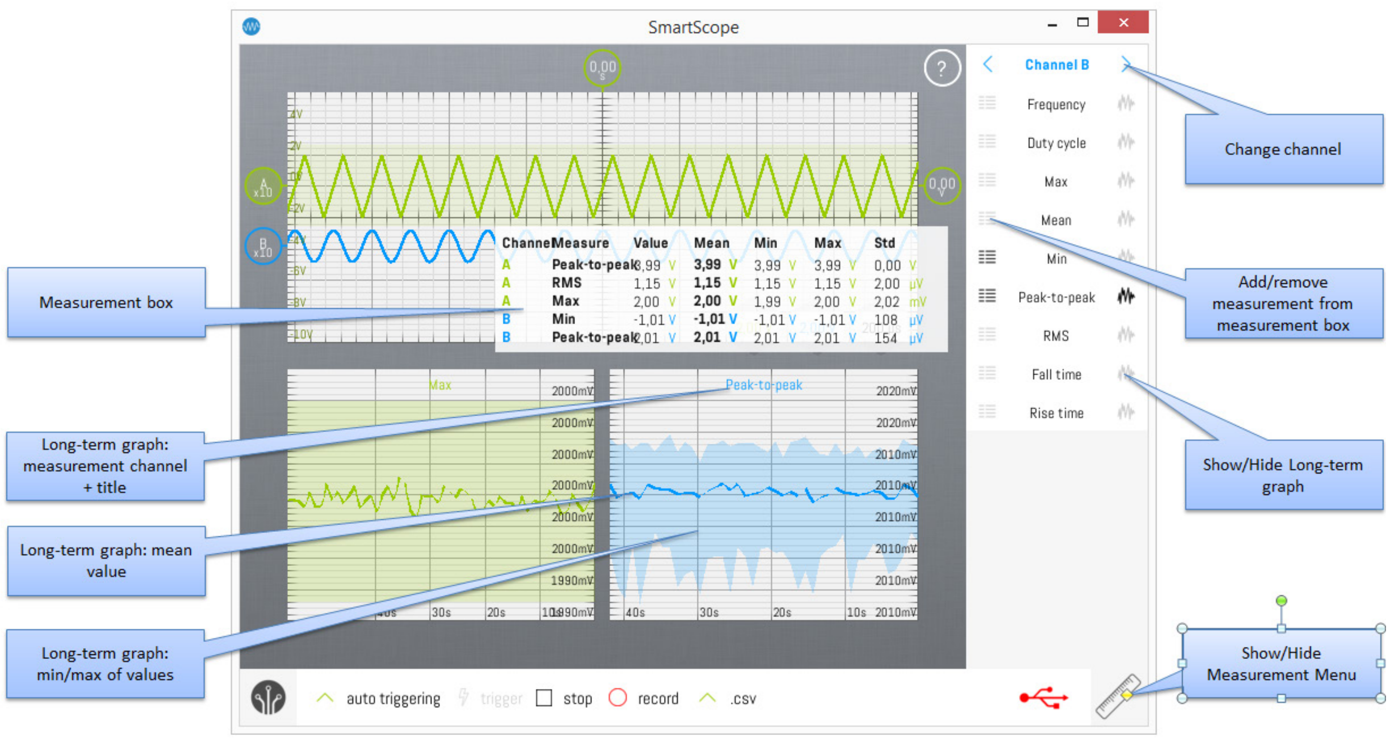


Channel	Measure	Value	Mean	Min	Max	Std
A	Peak-to-peak	3,99 V	3,99 V	3,99 V	3,99 V	0,00 V
A	RMS	1,15 V	1,15 V	1,15 V	1,15 V	2,00 µV
A	Max	2,00 V	2,00 V	1,99 V	2,00 V	2,02 mV
B	Min	-1,01 V	-1,01 V	-1,01 V	-1,01 V	108 µV
B	Peak-to-peak	2,01 V	2,01 V	2,01 V	2,01 V	154 µV

Contents

1 Overview

The main GUI elements of the measurement subsystem are shown below, which will be used throughout the remainder of this page:


SmartScope
Channel B
Frequency
Duty cycle
Max
Mean
Min
Peak-to-peak
RMS
Fall time
Rise time
Change channel
Add/remove measurement from measurement box
Show/Hide Long-term graph
Show/Hide Measurement Menu
Measurement box
Long-term graph: measurement channel + title
Long-term graph: mean value
Long-term graph: min/max of values
Max
Peak-to-peak
auto triggering
trigger
stop
record
.csv

Channel	Measure	Value	Mean	Min	Max	Std
A	Peak-to-peak	3,99 V	**3,99 V**	3,99 V	3,99 V	0,00 V
A	RMS	1,15 V	**1,15 V**	1,15 V	1,15 V	2,00 µV
A	Max	2,00 V	**2,00 V**	1,99 V	2,00 V	2,02 mV
B	Min	-1,01 V	**-1,01 V**	-1,01 V	-1,01 V	108 µV
B	Peak-to-peak	2,01 V	**2,01 V**	2,01 V	2,01 V	154 µV

2 The Measurements menu

The Measurements menu is used to show/hide measurements. The Measurements menu is integrated in the right part of the GUI. To open or hide the Measurements menu, hit the Ruler button at the bottom-right of the screen.
Once opened, the Measurement menu shows at its top the channel, followed by the measurements which can be invoked for this channel. Cycle through the channels by tapping the '<' and '>' buttons next to the channel name.

3 Measurement box

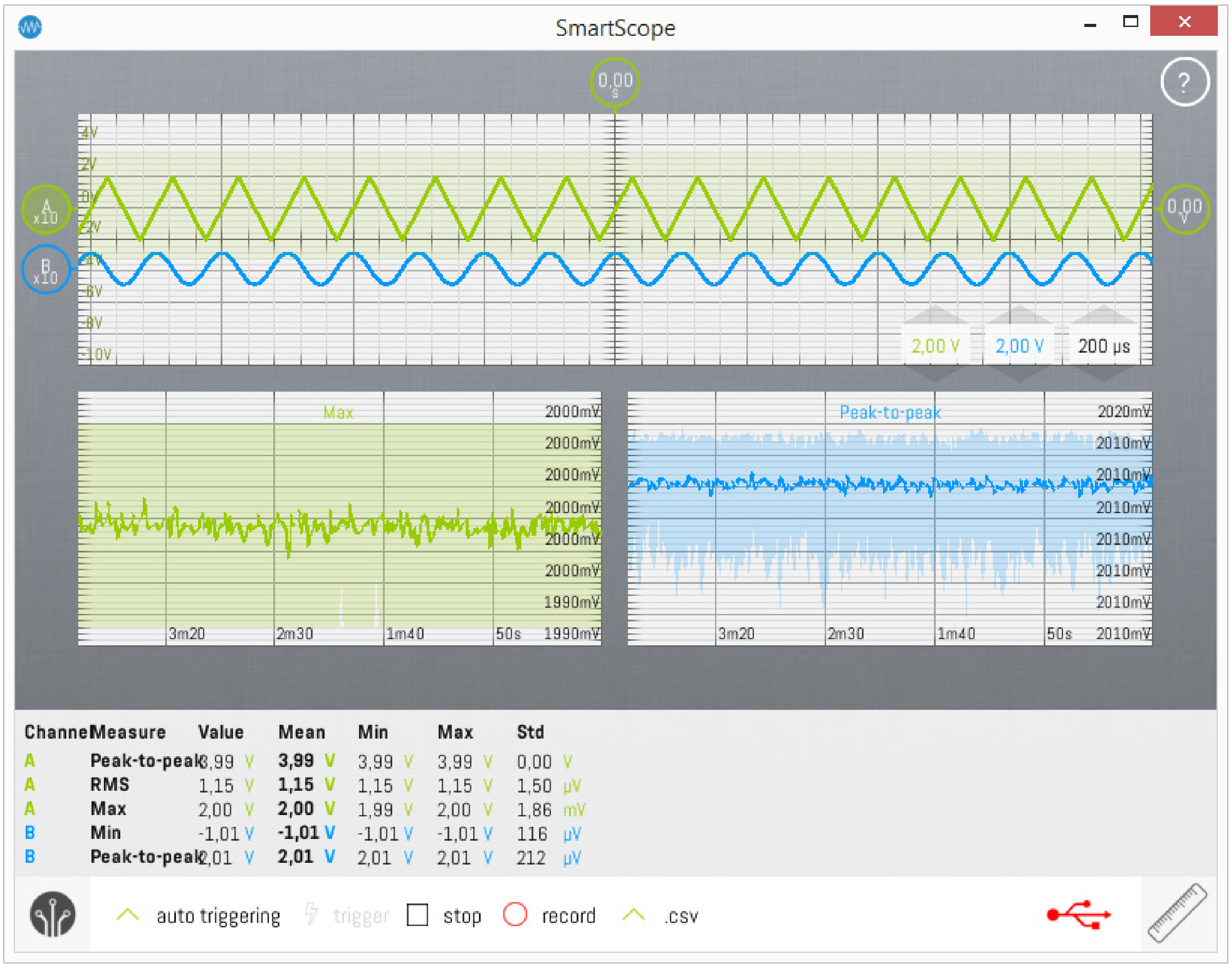
The measurement box contains the currently active measurements. The Value column displays the value calculated on each incoming wave. This value can vary very rapidly, and therefore the Mean column was added which shows the averaged value over 1 second. Added to this are the Min and Max columns, which show the minumum and maximal value of the measurement of said 1 second period. Finally, the Std column shows the standard deviation (~noise or variation) of this value over the 1 second interval.
The measurement box can be dragged around freely over the screen.

3.1 Showing/hiding the measurement box

By dragging the Measurement box out of the screen through the top or left side of the screen, it will be removed from the screen.
To show the measurement box again, simply open the Measurement menu. If the Measurement menu was already open, simply close and open the menu by tapping the ruler twice.

3.2 Docking the measurement box to the bottom of the screen

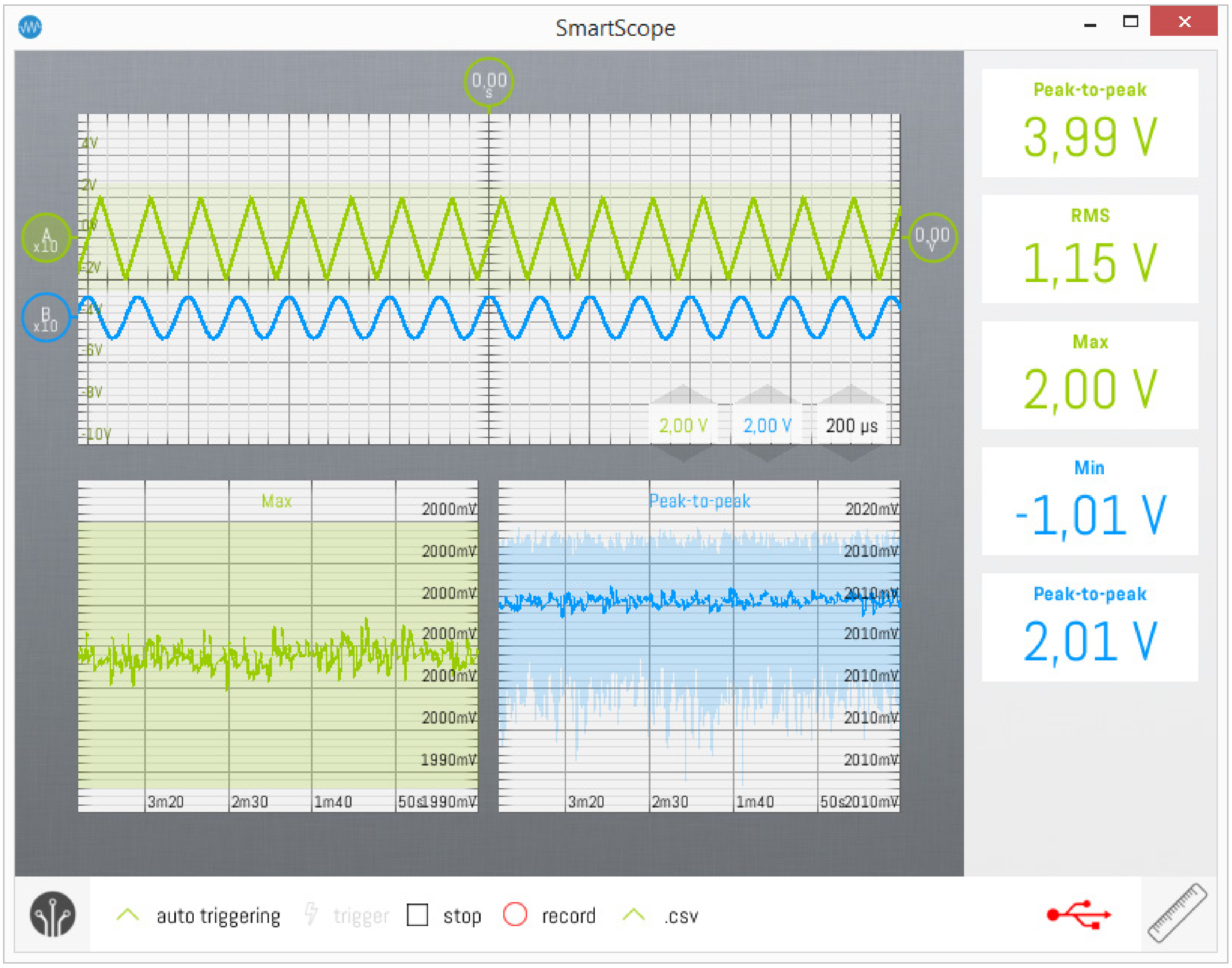
Typically the measurement box will be floating on top of your screen; and even though it is semi-transparant it can hide some parts you want to keep an eye on. Therefore, you can dock the measurement box to the bottom of the screen, which will make sure it no longer obscures other elements.



Channe	Measure	Value	Mean	Min	Max	Std
A	Peak-to-peak	3,99 V	**3,99 V**	3,99 V	3,99 V	0,00 V
A	RMS	1,15 V	**1,15 V**	1,15 V	1,15 V	1,50 µV
A	Max	2,00 V	**2,00 V**	1,99 V	2,00 V	1,86 mV
B	Min	-1,01 V	**-1,01 V**	-1,01 V	-1,01 V	116 µV
B	Peak-to-peak	2,01 V	**2,01 V**	2,01 V	2,01 V	212 µV

3.3 Docking the measurement box to the right side of the screen

Sometimes you're really only interested in the mean value of the measurement. For such cases, you can dock the measurement box to the right side of the screen, which will show the mean value in a much larger font. This can also be useful if you want to monitor a certain value from a distance.



Peak-to-peak
3,99 V

RMS
1,15 V

Max
2,00 V

Min
-1,01 V

Peak-to-peak
2,01 V

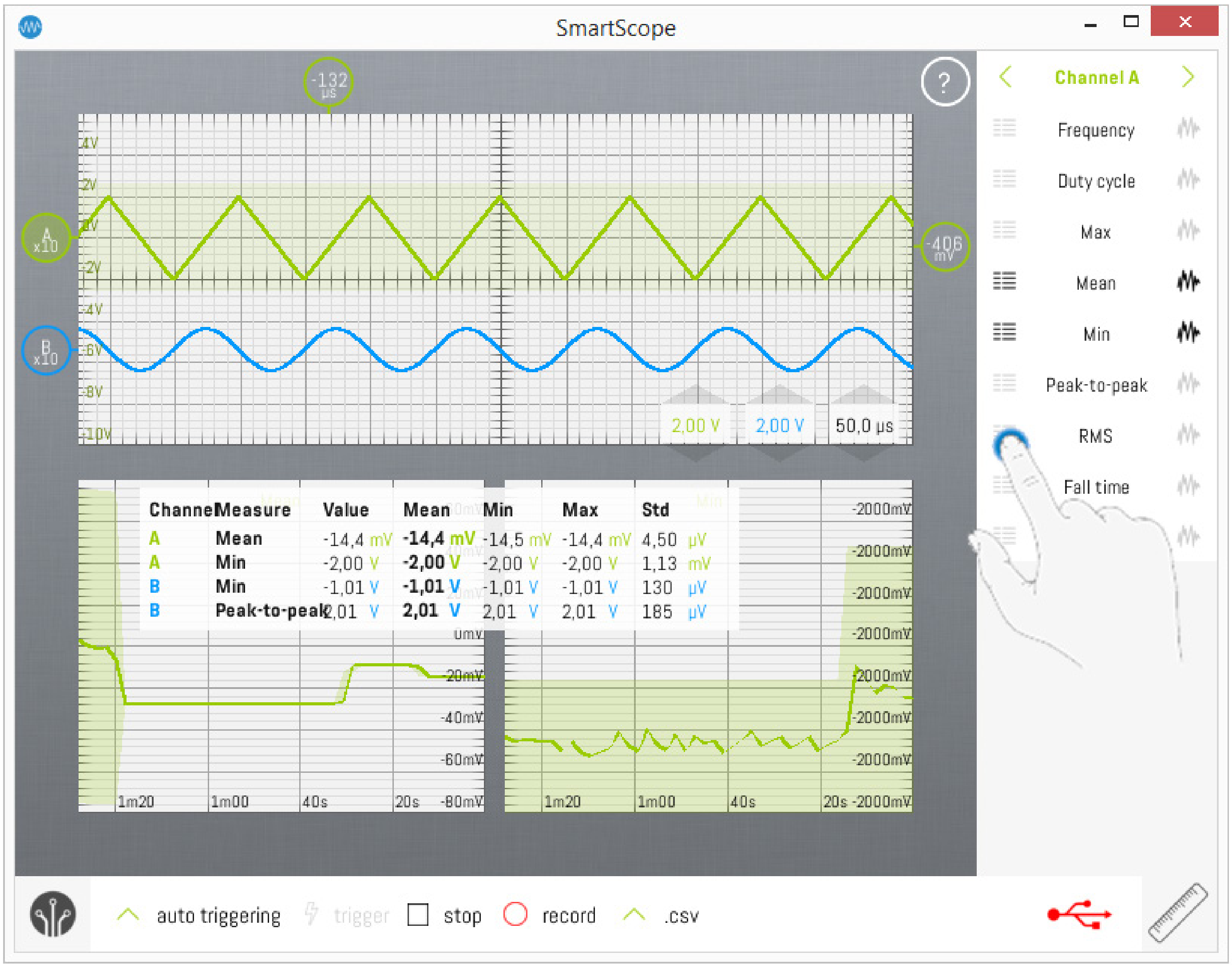
3.4 Adding/removing measurement to/from the measurement box

To add a measurement for a certain channel, follow these steps:

- Open the Measurement menu (by tapping the ruler at the bottom-right corner)
- Cycle to the desired channel (by tapping the '<' or '>' bottons at the top-right of the screen)
- Find your measurement, and hit the list icon to the left of the measurement's name

This will darken the list icon, and add the measurement to the Measurement box.

To remove the measurement again, repeat the same procedure which will gray out the list icon, and remove the measurment from the Measurement box.



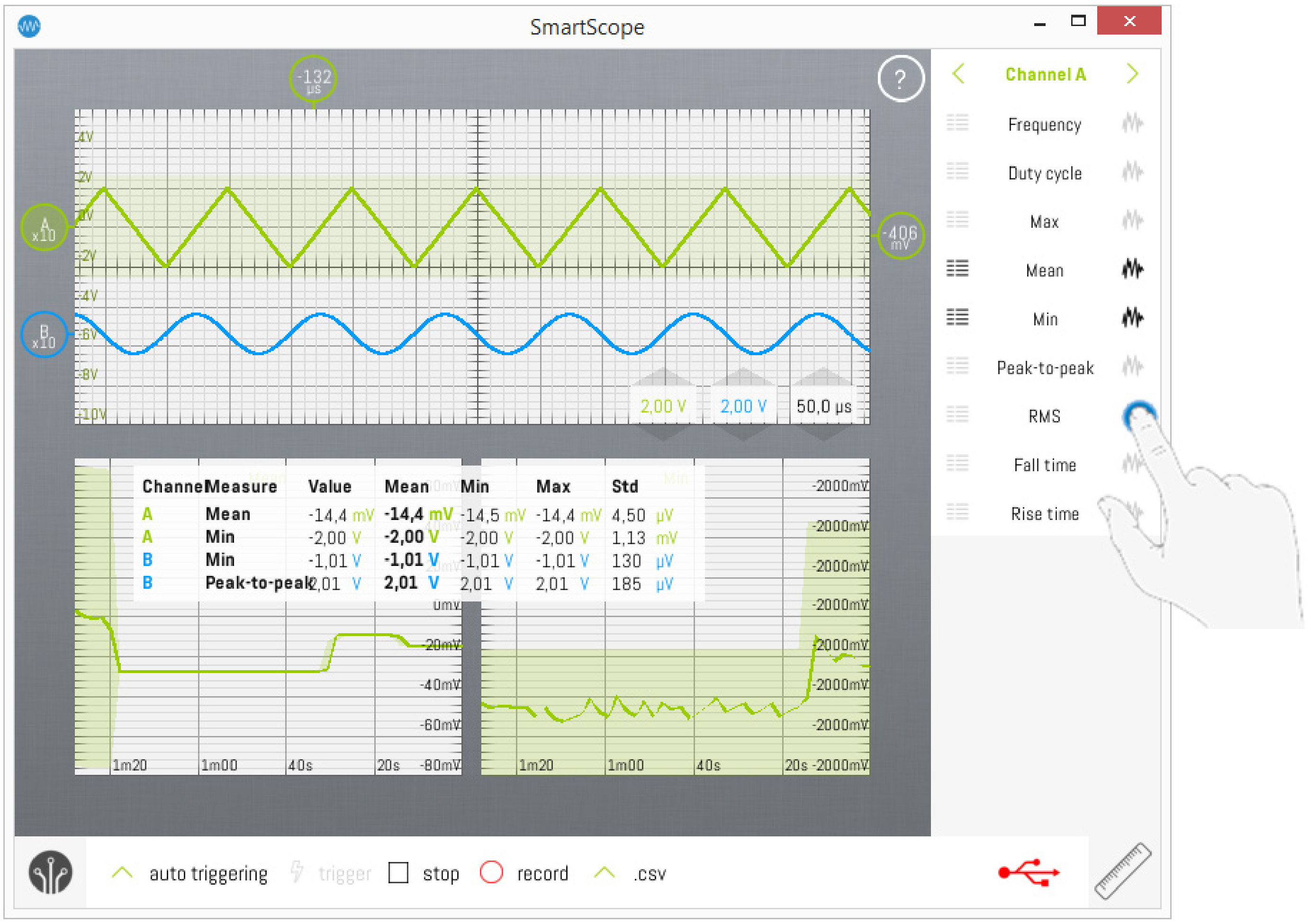
Channel	Measure	Value	Mean	Min	Max	Std
A	Mean	-14,4 mV	-14,4 mV	-14,5 mV	-14,4 mV	4,50 µV
A	Min	-2,00 V	-2,00 V	-2,00 V	-2,00 V	1,13 mV
B	Min	-1,01 V	-1,01 V	-1,01 V	-1,01 V	130 µV
B	Peak-to-peak	2,01 V	2,01 V	2,01 V	2,01 V	185 µV

4 Long-term measurement graphs

Sometimes you want to monitor a value over a long time. Using the long-term measurement graphs, this can now be done from within the SmartScope app.
In order to add the long-term graph for a measurement, follow these steps:

- Open the Measurement menu (by tapping the ruler at the bottom-right corner)
- Cycle to the desired channel (by tapping the '<' or '>' bottons at the top-right of the screen)
- Find your measurement, and hit the wave icon to the right of the measurement's name

This will darken the wave icon, and add the long-term graph of the measurement to the bottom of the main screen.



Channel	Measure	Value	Mean	Min	Max	Std
A	Mean	-14,4 mV	-14,4 mV	-14,5 mV	-14,4 mV	4,50 µV
A	Min	-2,00 V	-2,00 V	-2,00 V	-2,00 V	1,13 mV
B	Min	-1,01 V	-1,01 V	-1,01 V	-1,01 V	130 µV
B	Peak-to-peak	2,01 V	2,01 V	2,01 V	2,01 V	185 µV

Each long-term graph presents the following information:

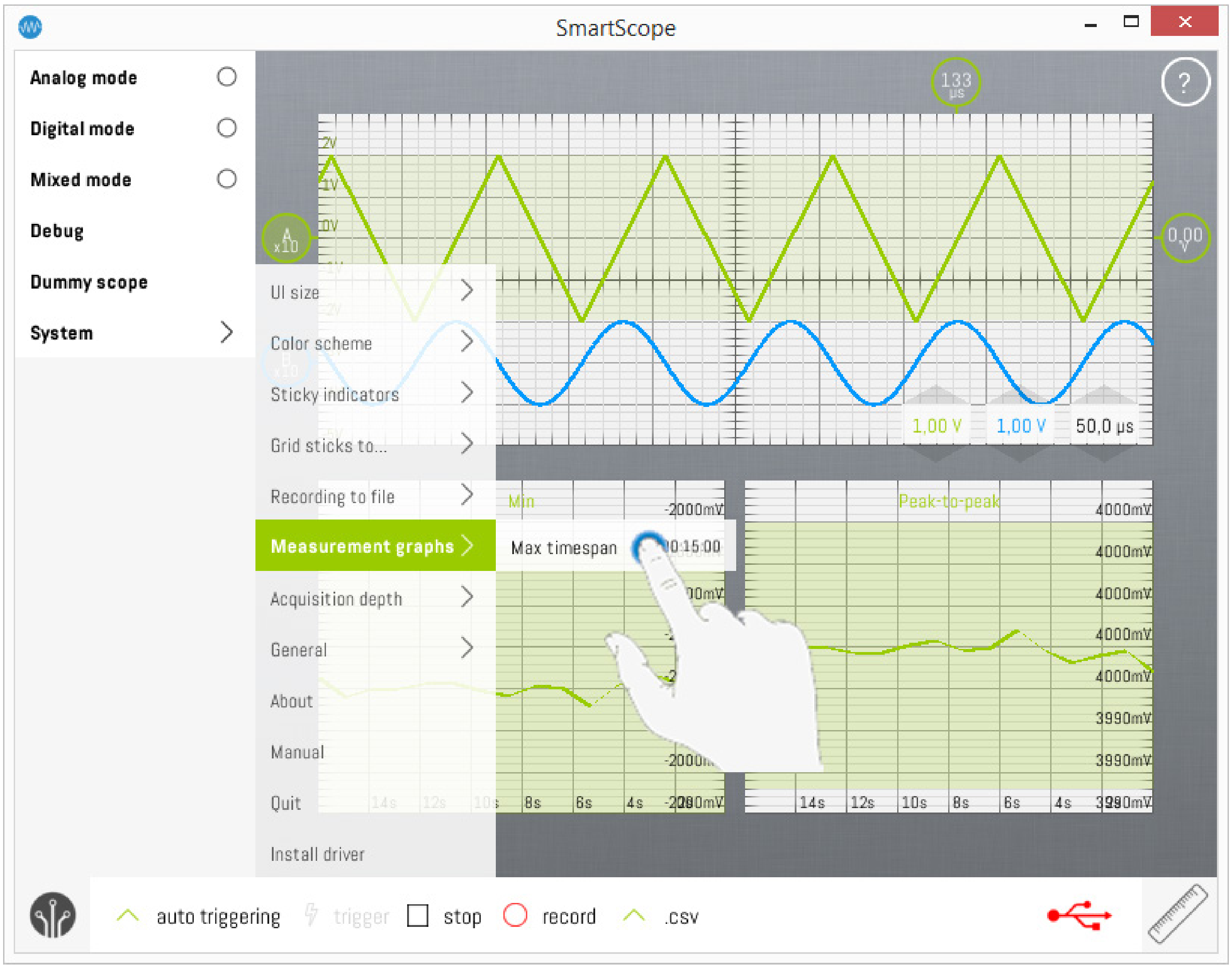
- Top-center: measurement title + color of the source channel
- x-axis: time axis (see below)
- Main trace: Mean value of that measurement
- Colored background behind trace: for each time-interval only the Mean value is shown on the main trace; but sometimes you need to follow the outliers as well. To this end, the colored background represent the Min and Max values of each interval.

4.1 Extending the timebase of the long-term channel graphs

By default up till 15 minutes of data will be shown on each long-term graph. This can be increased to any length you prefer, but keep in mind the lenght of the interval after which a new datapoint will be added to the graphs will increase. To do so, follow these steps:

- Open the main menu
- Go to System -> Measurement graphs and tap on the current timespan
- Enter any timespan you want through the keypad

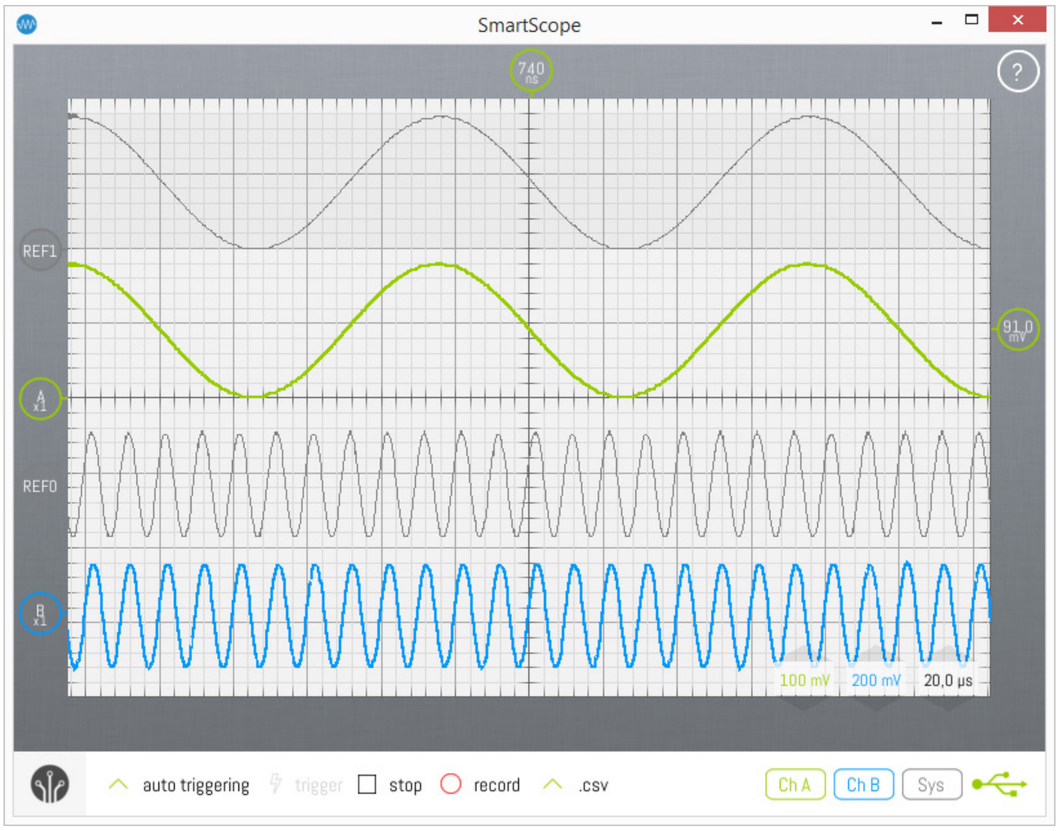
!!Warning: changing the timespan of the long-term graphs will clear all data of all active measurements!!



This page was last modified on 16 September 2017, at 23:17.

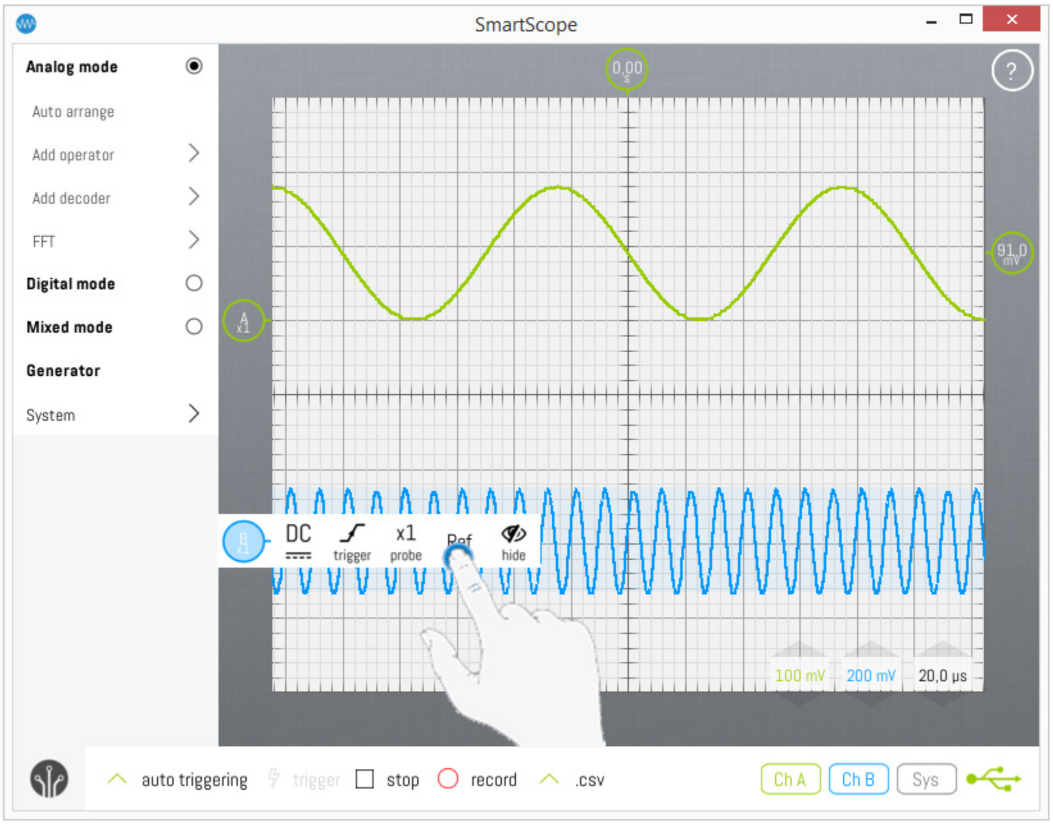
Reference waves

In some cases, you simply want to freeze a copy of a waveform to your screen. This can be done through the Reference wave.



Adding a reference wave

In order to duplicate an existing wave into a Reference wave, simply tap that wave's indicator on the left of the screen, and select the 'Ref' option, as shown below:



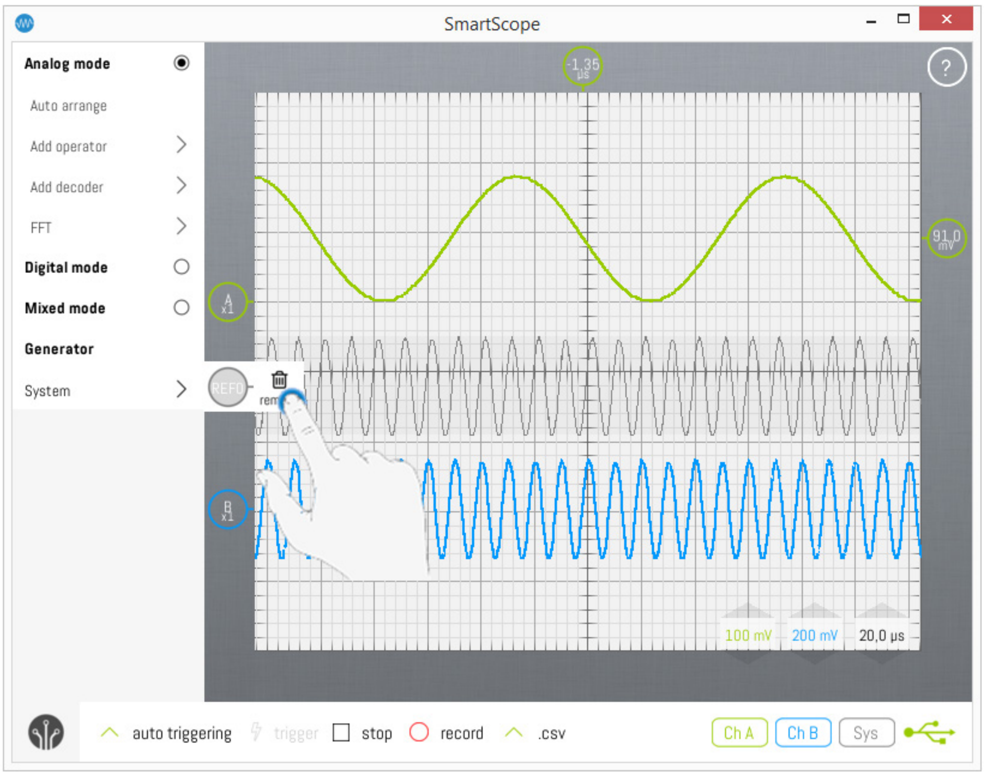
Note: you can add as many reference waves as you want.

Adjusting a reference wave

Since the idea of a Reference wave is to keep it steady on the screen, it is not possible to scale its voltage nor time base. You can, however, freely adjust the vertical position of a Reference wave. To do this, simply drag the wave or its indicator.

Deleting a reference wave

When you want to remove a Reference wave, simply tap its indicator on the left of the screen, and select Remove, as shown below:



Arbitrary Waveform Generator (AWG)

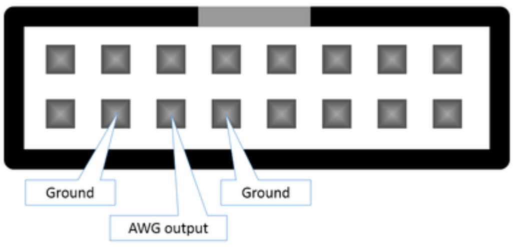
!!! NOTE: The Generator menu entry will only show up when a genuine SmartScope has been attached !!!

The SmartScope has an Arbitrary waveform generator, capable of generating signals between the [0V, 3.3V] voltage range at a sample rate of 100MS/s.

Contents

1 AWG pin location

The signal generated by the AWG is presented on the 3rd-left pin on the bottom row of the AUX connector, as shown in the following image:



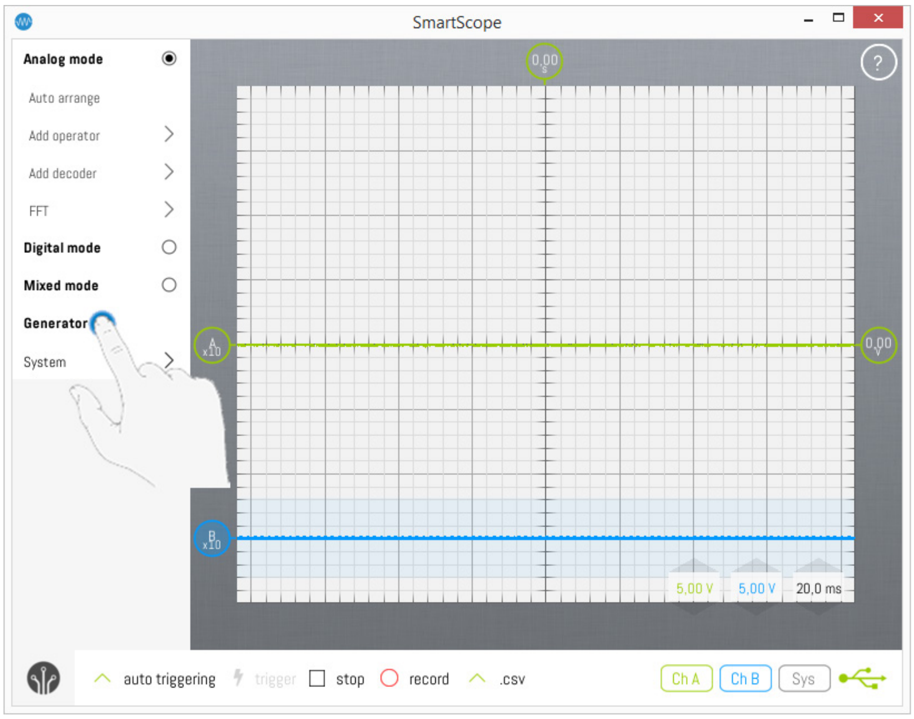
Please keep in mind that you always should bridge 2 wires between 2 separate devices. In this case:

- The AWG output signal
- The ground, so both devices have the same reference voltage. (0V means the same on both devices)

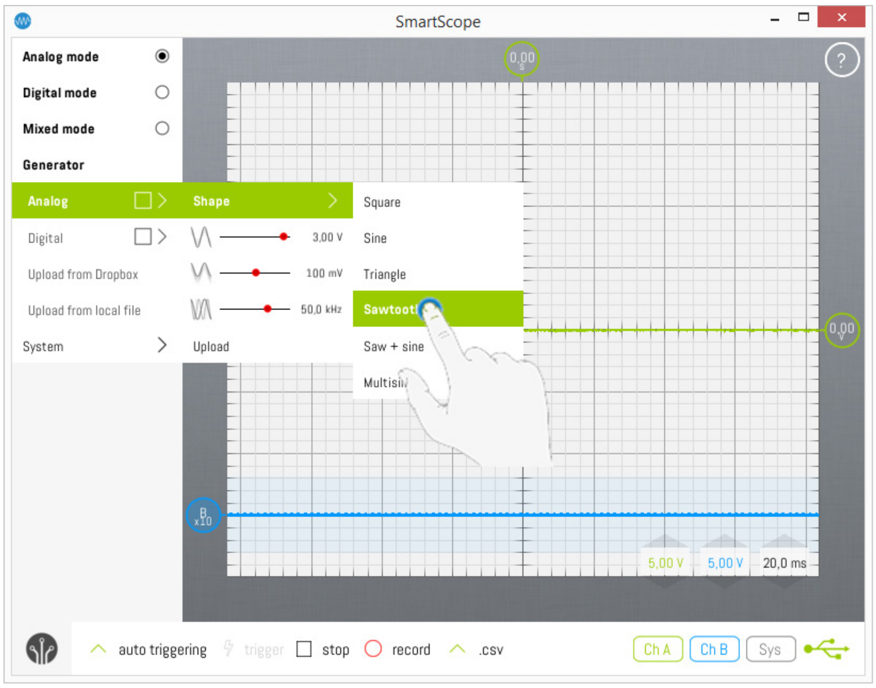
In the image above, you can see the AWG pin is surrounded by 2 ground pins, either of which you can use to connect to the other device.

2 Configuring the AWG using pre-defined waves

You can define your own waveforms (see below), or you can use one of the built-in signals which have been predefined in the SmartScope app. In order to do so, first open up the main menu by tapping on the LabNation logo at the bottom-left of the screen, and expand the Generator menu.



Next, open up the Analog submenu and select the type of predefined waveform you want the AWG to generate:



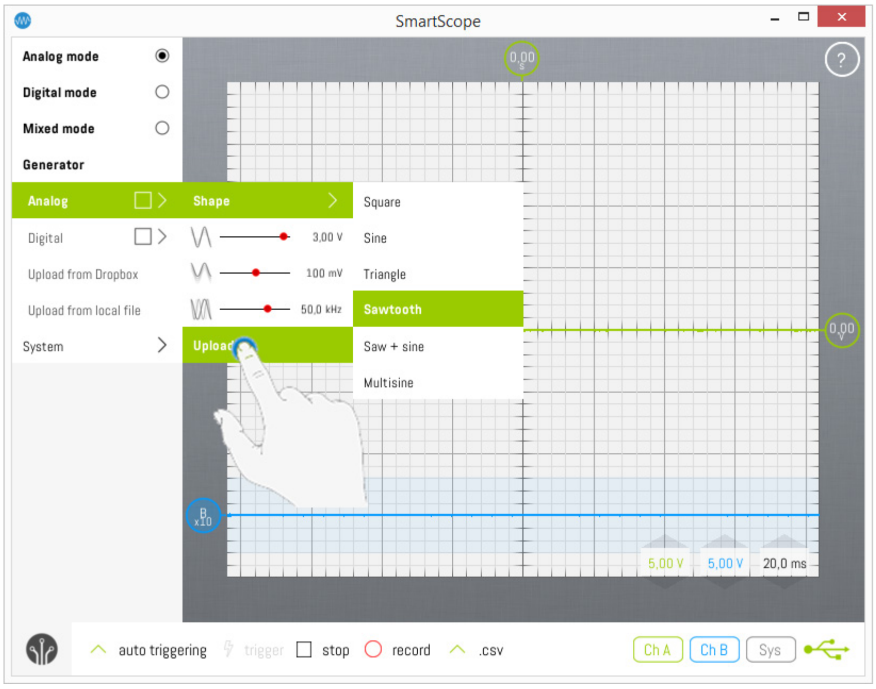
Configure the amplitude, offset and frequency of the wave to be generated.
TIP: Double-tapping the value will bring up the Numpad, which makes it much easier to specify accurate values.

NOTE: At this point the AWG output is not yet active. You have to use the 'Upload function' button and check the Analog checkbox to activate the output (see further).

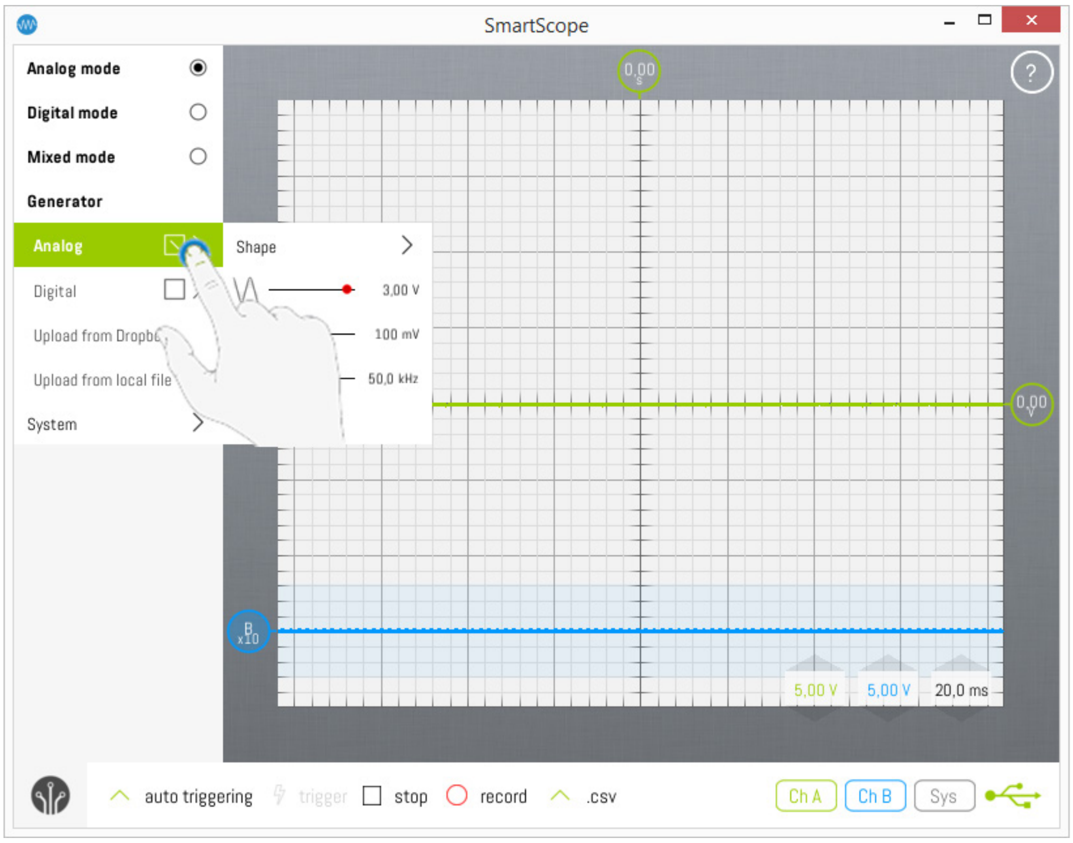
The AWG is configurable within these limits:

Parameter	Minimum value	Maximum value
Amplitude	0V	3.3V
Offset	0V	3.3V (max amplitude will decrease accordingly!)
Frequency	191Hz	781kHz

When you're done, hit the 'Upload' button, which will transfer the data to the SmartScope the AWG output. The Upload button will be highlighted while the data is being sent to the SmartScope.
NOTE: At this point the AWG output is not yet active. You still have to check the Analog checkbox to activate the output (see further).



With everything prepared, make sure to check the checkbox next to the Analog entry to enable the AWG output buffer:



3 Configuring the AWG using csv files

IMPORTANT: The app expects the CSV file to use a **semicolon** (;) as a field separator and a **comma** (,) as decimal symbol. Download one of the samples below to make sure your CSV works.

NOTE1: The CSV file **may** contain more than the number of samples specified in the parameters. Further samples will simply be ignored

NOTE2: This only works with a SmartScope connected. Otherwise, the side menu won't contain the Generator entry If you haven't used dropbox with the SmartScope

1. Tap sidemenu > Generator > Analog > Upload from dropbox
2. The app will tell you it doesn't have permission to dropbox and ask for it by sending you off to the dropbox website
3. Grant access and return to the app
4. The app now creates the AWG folder and will inform you that this new folder is empty.

1. Generate a CSV file using the AWG excel worksheet. A sample CSV can be found for a sine and block wave.
2. Drop your CSV file in the AWG folder (<dropbox>/**Apps**/LabNation SmartScope/AWG) using a file manager
3. In the app, tap sidemenu > AWG > Upload from dropbox
4. You should now be able to choose the CSV file

3.1 Under the hood

The AWG is driven by a 100MHz clock, reading out a memory containing *up to* 2048 samples.

3.2 Define arbitraty waveform in Excel (.csv file upload)

Parameter	Range	Description
DataIsBytes	0 or 1	Indicates if the data should be interpreted as bytes (for the 4 digital outputs). If '0', it will be interpreted as voltages (for the analog output).
Samples	1-2048	The number of samples used. The AWG loops on these samples. (< 0.0.8.4 had a minimum of 128 samples)
SampleStretch	0-255	The number of cycles to repeat each sample
BeginData		Field to indicate that from here on the sample voltage levels follow

The AWG excel worksheet contains examples of how to compute these parameters.

3.2.1 Sample CSV

```
Value;Field;Description
8;Samples;The number of samples to use (later samples are ignored)
244;SampleStretch;The number of times to repeat a sample
0,1;BeginData;Data begins here
0,2;;
0,3;;
0,4;;
0,3;;
0,2;;
0,1;;
0;;
```

In case you want to upload data as bytes (i.e. for digital output)

```
Value;Field;Description
1;DataIsBytes;Set to 1 to interpret the data as bytes, otherwise interpreted as voltages
8;Samples;The number of samples to use (later samples are ignored)
244;SampleStretch;The number of times to repeat a sample
0;BeginData;Data begins here
1;;
2;;
4;;
8;;
4;;
2;;
1;;
```

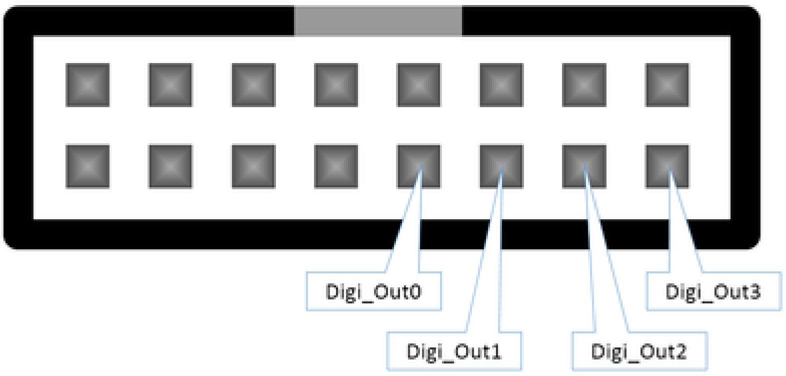
Digital Waveform Generator

The SmartScope comes with 4 dedicated digital output channels, each capable of generating up to 100MS/s with sharp rise and fall times of 2ns.
This allows square waves of up to 50MHz to be generated.
You can use the default app to generate some general signals, or upload your own csv file to generate any repeating sequence of up to 2048 samples.

Contents

1 Output pin locations

The 4 output channels are located on the AUX port, at the back of the SmartScope.



2 Output voltage levels

As output voltage levels, you can select between 2 standards:

- 0V - 3V
- 0V - 5V

These voltages are selectable for an entire waveform and for all 4 channels together, not per sample nor per channel.
In order to select the voltage, go to Menu -> Generator -> Digital -> Voltage level

3 Generating pre-defined waves

You can generate the predefined digital waves exactly the same way as desribed in Arbitrary Waveform Generator (AWG). Just make sure you have checked the checkbox in the Digital menu entry, to enable the digital output.

4 Define arbitrary waveforms in Excel (.csv file upload)

Parameter	Range	Description
DataIsBytes	0 or 1	Indicates if the data should be interpreted as bytes (for the 4 digital outputs). If '0', it will be interpreted as voltages (for the analog output).
Samples	1-2048	The number of samples used. The AWG loops on these samples. (< 0.0.8.4 had a minimum of 128 samples)
SampleStretch	0-255	The number of cycles to repeat each sample. If 0, sample update rate is 100MHz. If n, sample update rate is 100/n MHz.
BeginData		Field to indicate that from here on the sample voltage levels follow

Each row defines a sample as byte value, where the 4 LSB define the values of the 4 output pins.

CsvValue	Output3	Output2	Output1	Output0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0
etc	etc	etc	etc	etc

Sample digital output CSV

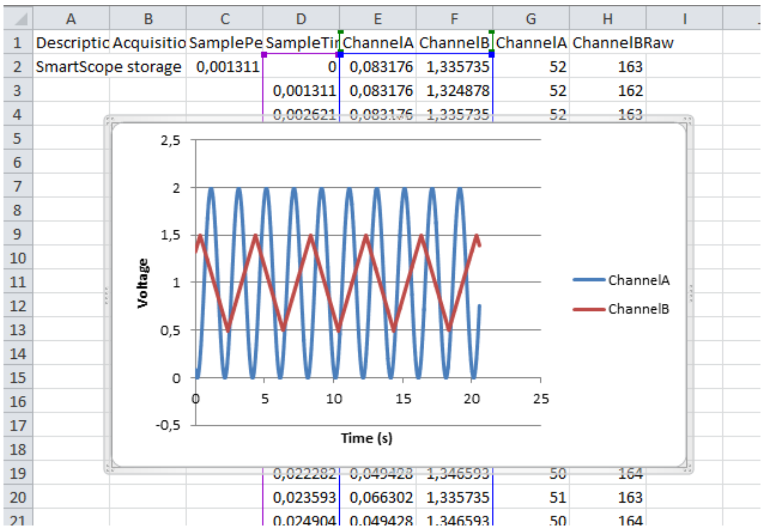
```
Value;Field;Description
1;DataIsBytes;Set to 1 to interpret the data as bytes, otherwise interpreted as voltages
8;Samples;The number of samples to use (later samples are ignored)
244;SampleStretch;The number of times to repeat a sample
0;BeginData;Data begins here
1;;
2;;
4;;
8;;
4;;
2;;
1;;
```

This page was last modified on 26 April 2017, at 15:20.

Recording data to disk

The SmartScope can record all acquired data to readable comma separated values (.csv) files which can be opened with eg Excel, or to much more compact .mat file which can be opened using Matlab or Octave.

The first secion of this page discusses how to record your acquisitions, while the second section deals with manipulating and visualizing the data recorded to disk.

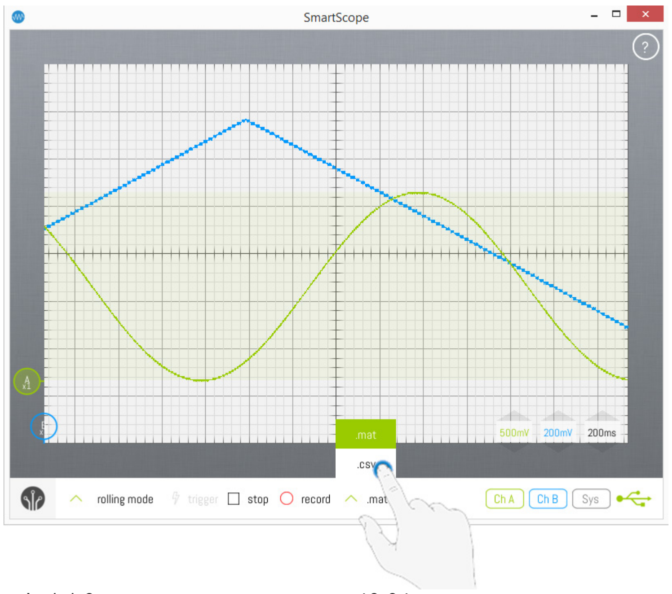


	A	B	C	D	E	F	G	H	I
1	Descriptic	Acquisitio	SamplePe	SampleTir	ChannelA	ChannelB	ChannelA	ChannelBRaw	
2	SmartScope storage		0,001311	0	0,083176	1,335735	52	163	
3				0,001311	0,083176	1,324878	52	162	
4				0,002621	0,083176	1,335735	52	163	
19				0,022282	0,049428	1,346593	50	164	
20				0,023593	0,066302	1,335735	51	163	
21				0,024904	0,049428	1,346593	50	164	

Contents

1 Selecting the output file type

Start by selecting your output file type at the bottom of the main screen.
Opening and visualizing the contents of these files will be handled further down this document.



2 SmartScope recording modes

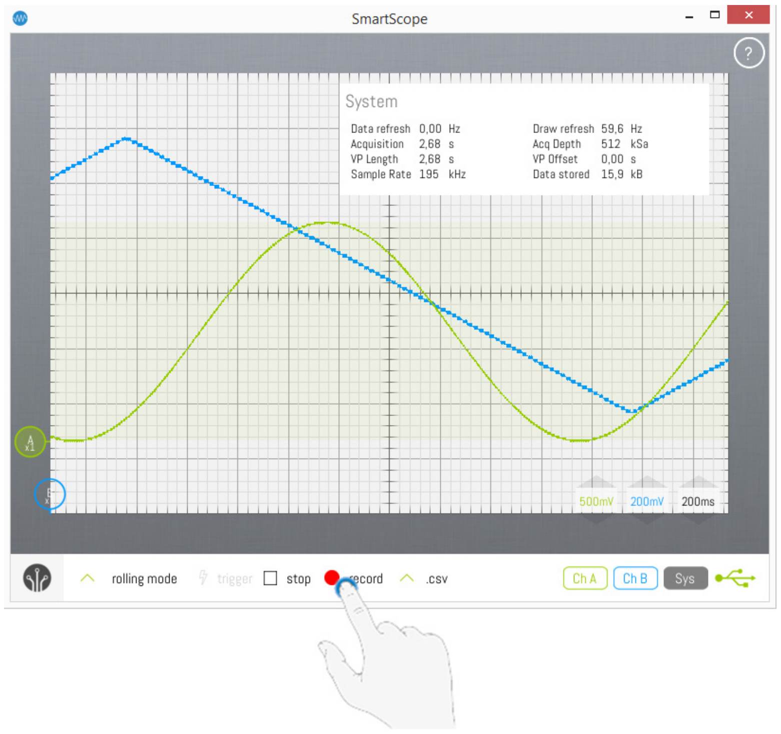
You can choose between 4 ways of recording your data, each of them explained further on:

- Slow signals, long timespan recording (rolling mode)
- Fast signals, record all data (non-rolling mode)
- Fast signals, record every X seconds (non-rolling mode)
- Full RAM contents (non-rolling mode, single acquisition)

2.1 Slow signals, long timespan recording (rolling mode)

In case you want to record slow signals over an extended amount of time, you simply zoom out on the timescale until the SmartScope switches to Rolling mode, and hit the Record button. You should notice the following while recording:

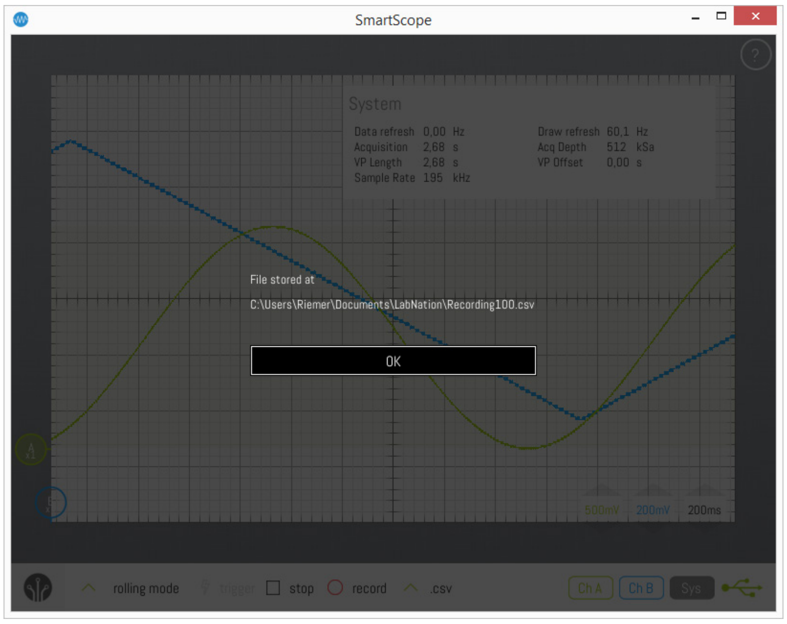
- The Record button is now blinking
- The System measurement box indicates the amount of data stored
- Most GUI elements have been disabled (as otherwise this would invalidate the recording file)



System

Data refresh 0,00 Hz
Acquisition 2,68 s
VP Length 2,68 s
Sample Rate 195 kHz
Draw refresh 59,6 Hz
Acq Depth 512 kSa
VP Offset 0,00 s
Data stored 15,9 kB

While recording in rolling mode, all channels will be sampled at around 700Hz, and each sample will be stored to file. When you've finished recording what you need, simply hit the record button again, and a dialog will show you where the file has been stored.



System

Data refresh 0,00 Hz
Acquisition 2,68 s
VP Length 2,68 s
Sample Rate 195 kHz
Draw refresh 60,1 Hz
Acq Depth 512 kSa
VP Offset 0,00 s

File stored at
C:\Users\Riemer\Documents\LabNation\Recording100.csv

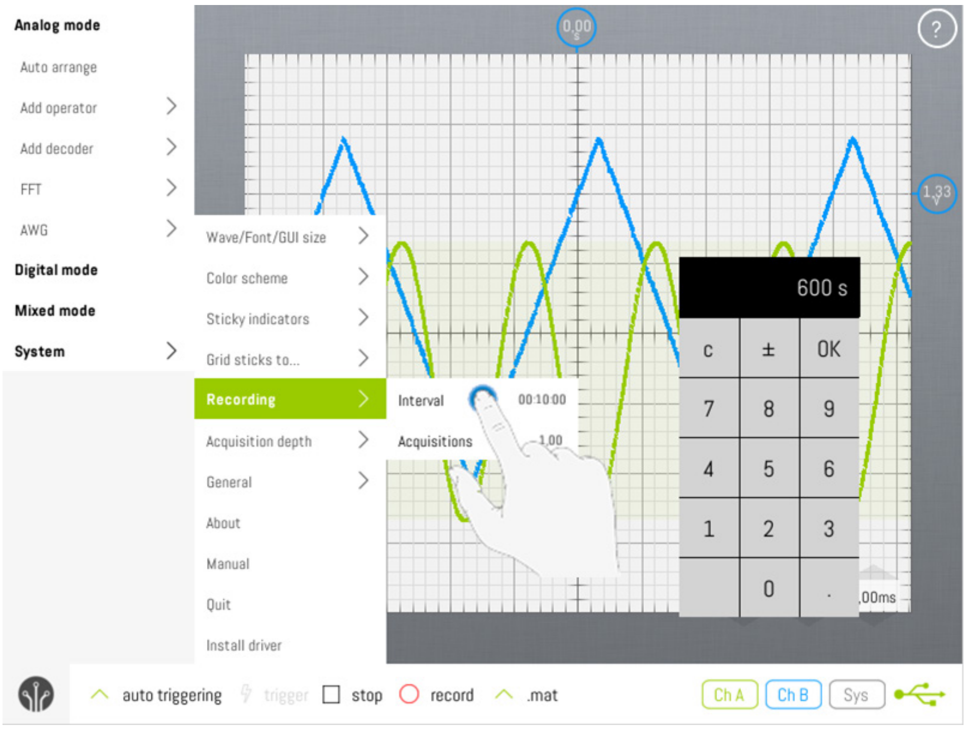
2.2 Fast signals, record all data (non-rolling mode)

In case you want to record higher-frequency signals, you'll typically be out of rolling mode. With sufficiently fast timescale, there can be more than 100 screen updates every second. Every incoming voltage can be saved to disk, simply by hitting the Record button. As long as you're recording, all incoming data is both stored to disk (hiPriority) and shown on the main graph (lowPriority).

Note: this means the final filesize can grow rapidly! Keep on eye on the 'Data stored' entry in the System measurement box. Especially .csv files can grow large, making them very slow to open/handle in Excel.

2.3 Fast signals, record every X seconds (non-rolling mode)

The previous scenarios don't allow you to record for really long timespans (eg hours or days), simply because the amount of data saved to disk would result in huge files. This can be done, but you'll face trouble when opening them, especially with Excel which is really slow in handling large files. Instead, the SmartScope allows you to save only 1 recording every X seconds. Or you can also save a burst of Y recordings every X seconds. In order to do this, go to Menu -> System -> Recording, from which you can both set the Interval in seconds (X in the text above), as well as the number of Acquisitions to store in one burst (Y in the text above).



Now when you press the Record button, you'll see that the 'Data stored' property of the Systems measurement box only increases at each interval you specified above. As a result, the file on disk will only contain the bursts at intervals you specify, allowing you to record over very large timespans and still keep the file very small.

'''Note:''' this doesn't work in '''rolling mode'''. However, while in rolling mode you can always set the the Trigger mode to 'auto trigger'. This gives a visually less appealing result, but allows you to record with a specified interval.

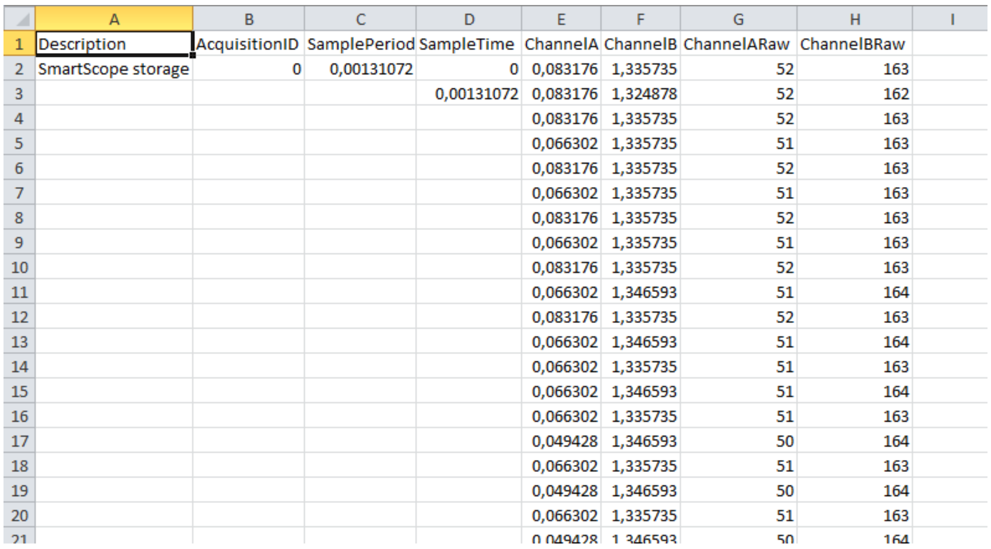
2.4 Full RAM contents (single acquisition only)

In all other modes above, all data shown on the screen (and potentially a bit more to the left and right) will be recorded to disk, which typically means 2048 samples for each channel for each acquisition. If you want to store the contents of the full RAM, this is also possible: simply stop the acquisition and wait till the full RAM contents has been downloaded (indicated by the green progress bar in the Panorama). Once this is complete, hit the Record button. You will be presented with the dialog below, asking you if you want to store the current data. When you click OK, the full contents of the RAM will be stored to disk.
Important: You have to wait until the entire contents of the RAM has been transferred (see previous line). When you hit the Record button before this transfer is complete, only the Viewport data (2048 samples) will be written to file.
Note: Excel has a limit of max 1 million rows, so if you're using csv+Excel make sure you keep the RAM depth to the default setting of 512k.

3 Working with recorded .csv files

3.1 Examining a rolling .csv file data using Excel

Navigate to the recorded file. In case MS Excel is installed on your system, simply double-click the file and its raw contents will be displayed in Excel.



	A	B	C	D	E	F	G	H	I
1	Description	AcquisitionID	SamplePeriod	SampleTime	ChannelA	ChannelB	ChannelARaw	ChannelBRaw	
2	SmartScope storage	0	0,00131072	0	0,083176	1,335735	52	163	
3				0,00131072	0,083176	1,324878	52	162	
4					0,083176	1,335735	52	163	
5					0,066302	1,335735	51	163	
6					0,083176	1,335735	52	163	
7					0,066302	1,335735	51	163	
8					0,083176	1,335735	52	163	
9					0,066302	1,335735	51	163	
10					0,083176	1,335735	52	163	
11					0,066302	1,346593	51	164	
12					0,083176	1,335735	52	163	
13					0,066302	1,346593	51	164	
14					0,066302	1,335735	51	163	
15					0,066302	1,346593	51	164	
16					0,066302	1,335735	51	163	
17					0,049428	1,346593	50	164	
18					0,066302	1,335735	51	163	
19					0,049428	1,346593	50	164	
20					0,066302	1,335735	51	163	
21					0,049428	1,346593	50	164	

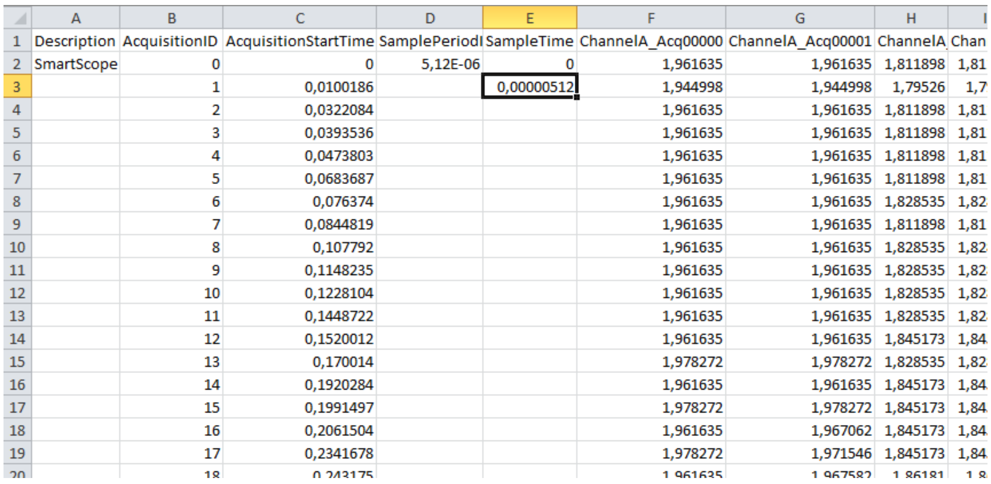
You'll notice the following columns:

- Description: field containing date and time when the file was created
- AcquisitionID: not used for rolling mode
- SamplePeriod: the amount of seconds between each recorded sample
- SampleTime: useful for creating time axis, see 'Creating the time axis' below
- ChannelA (B): the voltage of ChannelA(B)
- ChannelARaw (B): the unprocessed bytevalue of channel A (B) returned by the hardware

In most cases, you'll be interested in ChannelA/B columns, as these contain the voltages of both channels. Each row contains a new acquisition, with the time between each acquisition specified in the SamplePeriod column.

3.2 Examining a non-rolling .csv file data using Excel

In case your csv file was recorded while not in rolling mode (to record higher-frequency signals), a lot more data will be present in your file. Basically, each waveform displayed while recording will be saved as a column of up to 2048 samples. Simply double-click on the file to open it with Excel.



	A	B	C	D	E	F	G	H	I
1	Description	AcquisitionID	AcquisitionStartTime	SamplePeriodI	SampleTime	ChannelA_Acq00000	ChannelA_Acq00001	ChannelA	Chan
2	SmartScope	0	0	5,12E-06	0	1,961635	1,961635	1,811898	1,81
3		1	0,0100186		0,00000512	1,944998	1,944998	1,79526	1,7
4		2	0,0322084			1,961635	1,961635	1,811898	1,81
5		3	0,0393536			1,961635	1,961635	1,811898	1,81
6		4	0,0473803			1,961635	1,961635	1,811898	1,81
7		5	0,0683687			1,961635	1,961635	1,811898	1,81
8		6	0,076374			1,961635	1,961635	1,828535	1,82
9		7	0,0844819			1,961635	1,961635	1,811898	1,81
10		8	0,107792			1,961635	1,961635	1,828535	1,82
11		9	0,1148235			1,961635	1,961635	1,828535	1,82
12		10	0,1228104			1,961635	1,961635	1,828535	1,82
13		11	0,1448722			1,961635	1,961635	1,828535	1,82
14		12	0,1520012			1,961635	1,961635	1,845173	1,84
15		13	0,170014			1,978272	1,978272	1,828535	1,82
16		14	0,1920284			1,961635	1,961635	1,845173	1,84
17		15	0,1991497			1,978272	1,978272	1,845173	1,84
18		16	0,2061504			1,961635	1,967062	1,845173	1,84
19		17	0,2341678			1,978272	1,971546	1,845173	1,84
20		18	0,243175			1,961635	1,967582	1,86181	1,8

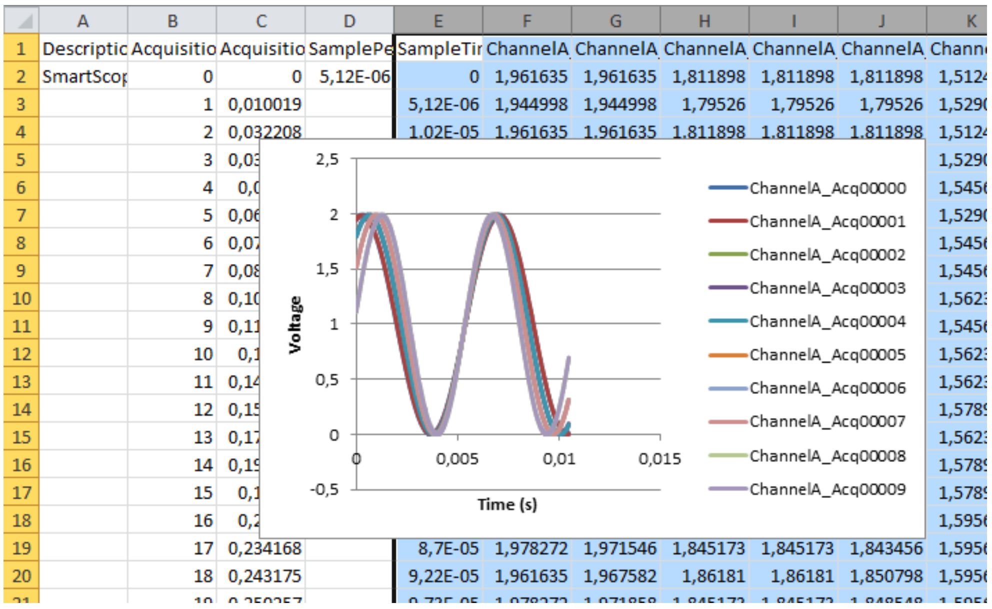
You'll notice the following columns:

- Description: field containing date and time when the file was created
- AcquisitionID & AcquisitionStartTime: The identier of each acquisition set of 2048 samples, together with the exact timestamp of first sample. This is useful for determining the time between 2 acquisition sets.
- SamplePeriod: the amount of seconds between each recorded sample (between each Excel row)
- SampleTime: useful for creating time axis, see 'Creating the time axis' below
- ChannelA_Acq00000 -> ChannelA_AcqXYZ: each column represents an acquisition of up to 2048 samples of this channel. The time between each sample (row) is given in the SamplePeriod column (see above). The

time between each acquisition set (column) can be derived from the AcquisitionID & AcquisitionStartTime columns (see above)

- Other data columns: each channel will have as many columns as the number of acquisitions recorded. This means unprocessed bytevalues, digital channels, decoders and analog operators will all have their columns.

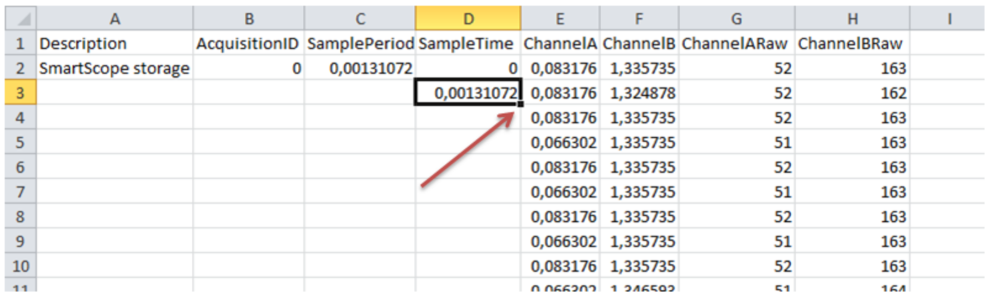
The image below shows an example of how a fastly moving signal can be visualized in Excel. See the next section on more information on how to convert the data discussed above into a graph.



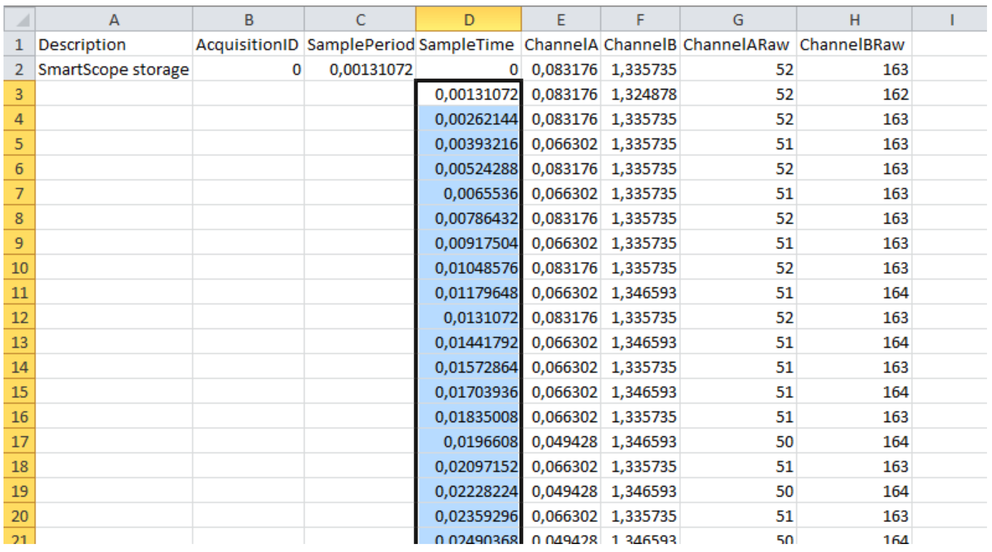
	A	B	C	D	E	F	G	H	I	J	K
1	Descriptic	Acquisitio	Acquisitio	SamplePe	SampleTir	ChannelA	ChannelA	ChannelA	ChannelA	ChannelA	Chann
2	SmartScop	0	0	5,12E-06	0	1,961635	1,961635	1,811898	1,811898	1,811898	1,512
3		1	0,010019		5,12E-06	1,944998	1,944998	1,79526	1,79526	1,79526	1,529
4		2	0,032208		1,02E-05	1,961635	1,961635	1,811898	1,811898	1,811898	1,512
5		3	0,03								1,529
6		4	0,0								1,545
7		5	0,06								1,529
8		6	0,07								1,545
9		7	0,08								1,545
10		8	0,10								1,562
11		9	0,11								1,545
12		10	0,1								1,562
13		11	0,14								1,562
14		12	0,15								1,578
15		13	0,17								1,562
16		14	0,19								1,578
17		15	0,1								1,578
18		16	0,2								1,595
19		17	0,234168		8,7E-05	1,978272	1,971546	1,845173	1,845173	1,843456	1,595
20		18	0,243175		9,22E-05	1,961635	1,967582	1,86181	1,86181	1,850798	1,595
21		19	0,250257		9,73E-05	1,978272	1,971858	1,845173	1,845173	1,848548	1,595

3.3 Creating the time axis - visualizing the data in Excel

Typically, you'll want to plot the voltages against the time. The file contains a hidden formula which lets you create the time-axes automatically: select field D3, and double-click on its bottom-right corner. This will make Excel fill the entire column with the timestamp of that acquisition!

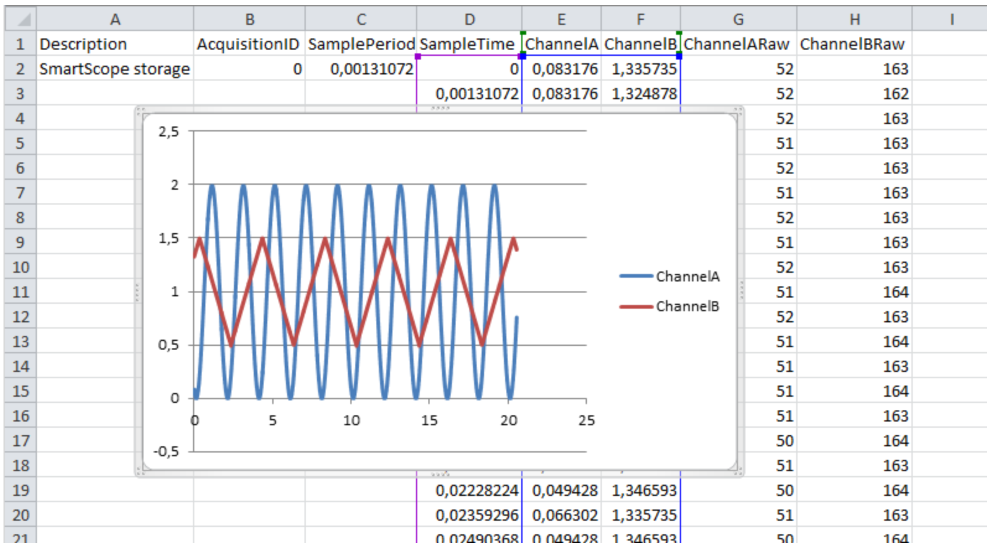


	A	B	C	D	E	F	G	H	I
1	Description	AcquisitionID	SamplePeriod	SampleTime	ChannelA	ChannelB	ChannelARaw	ChannelBRaw	
2	SmartScope storage	0	0,00131072	0	0,083176	1,335735	52	163	
3				0,00131072	0,083176	1,324878	52	162	
4					0,083176	1,335735	52	163	
5					0,066302	1,335735	51	163	
6					0,083176	1,335735	52	163	
7					0,066302	1,335735	51	163	
8					0,083176	1,335735	52	163	
9					0,066302	1,335735	51	163	
10					0,083176	1,335735	52	163	
11					0,066302	1,346593	51	164	



	A	B	C	D	E	F	G	H	I
1	Description	AcquisitionID	SamplePeriod	SampleTime	ChannelA	ChannelB	ChannelARaw	ChannelBRaw	
2	SmartScope storage	0	0,00131072	0	0,083176	1,335735	52	163	
3				0,00131072	0,083176	1,324878	52	162	
4				0,00262144	0,083176	1,335735	52	163	
5				0,00393216	0,066302	1,335735	51	163	
6				0,00524288	0,083176	1,335735	52	163	
7				0,0065536	0,066302	1,335735	51	163	
8				0,00786432	0,083176	1,335735	52	163	
9				0,00917504	0,066302	1,335735	51	163	
10				0,01048576	0,083176	1,335735	52	163	
11				0,01179648	0,066302	1,346593	51	164	
12				0,0131072	0,083176	1,335735	52	163	
13				0,01441792	0,066302	1,346593	51	164	
14				0,01572864	0,066302	1,335735	51	163	
15				0,01703936	0,066302	1,346593	51	164	
16				0,01835008	0,066302	1,335735	51	163	
17				0,0196608	0,049428	1,346593	50	164	
18				0,02097152	0,066302	1,335735	51	163	
19				0,02228224	0,049428	1,346593	50	164	
20				0,02359296	0,066302	1,335735	51	163	
21				0,02490368	0,049428	1,346593	50	164	

With the time-data and voltage-data present, you can use Excel basics to draw a graph. Simply select the time-data (column D) and voltage-data columns (E and F) by clicking and dragging from D to F on the column names, then go to Insert -> Graph -> Scatter -> Lines.



	A	B	C	D	E	F	G	H	I
1	Description	AcquisitionID	SamplePeriod	SampleTime	ChannelA	ChannelB	ChannelARaw	ChannelBRaw	
2	SmartScope storage	0	0,00131072	0	0,083176	1,335735	52	163	
3				0,00131072	0,083176	1,324878	52	162	
4							52	163	
5							51	163	
6							52	163	
7							51	163	
8							52	163	
9							51	163	
10							52	163	
11							51	164	
12							52	163	
13							51	164	
14							51	163	
15							51	164	
16							51	163	
17							50	164	
18							51	163	
19				0,02228224	0,049428	1,346593	50	164	
20				0,02359296	0,066302	1,335735	51	163	
21				0,02490368	0,049428	1,346593	50	164	

4 Working with recorded .mat files

-under construction-

This page was last modified on 2 March 2016, at 09:41.

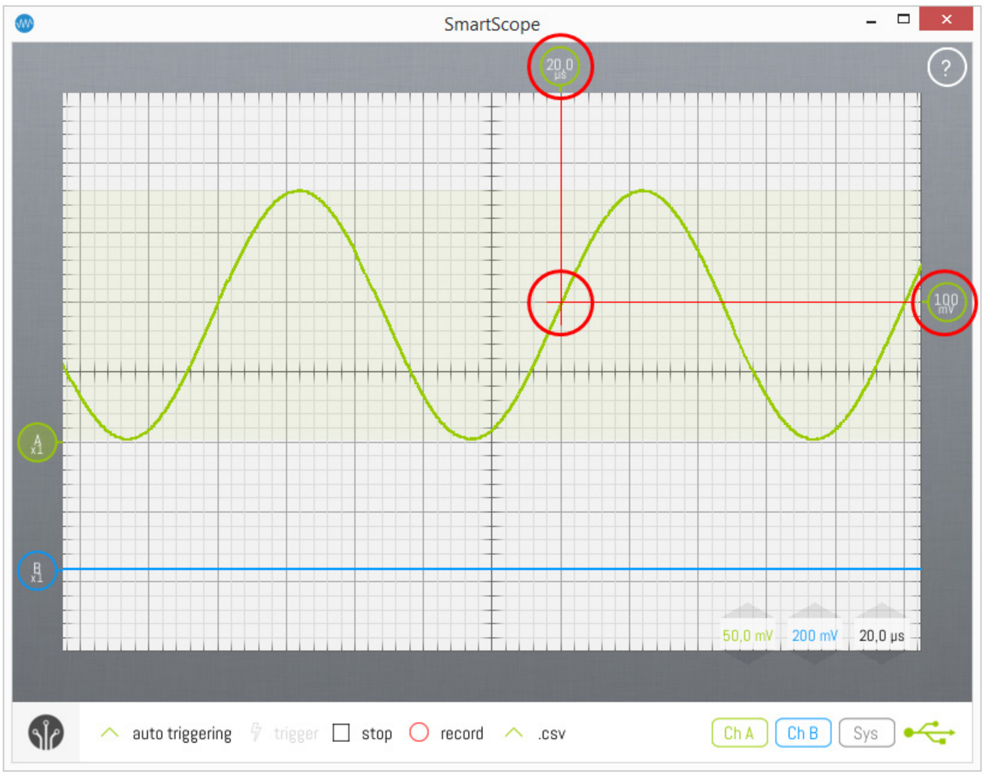
Extended functionality

Advanced triggering options

Contents

1 Triggering basics

Without triggering, the SmartScope would render each acquisition of a simple sine wave at a different timelocation. At ~150 acquisitions per second, this would make the sine wave appear all over the place. What you need, is the ability to define a certain voltage, and tell the SmartScope to position this voltage always at the same location on the screen. This is called triggering, and will result in a stable waveform on the screen.

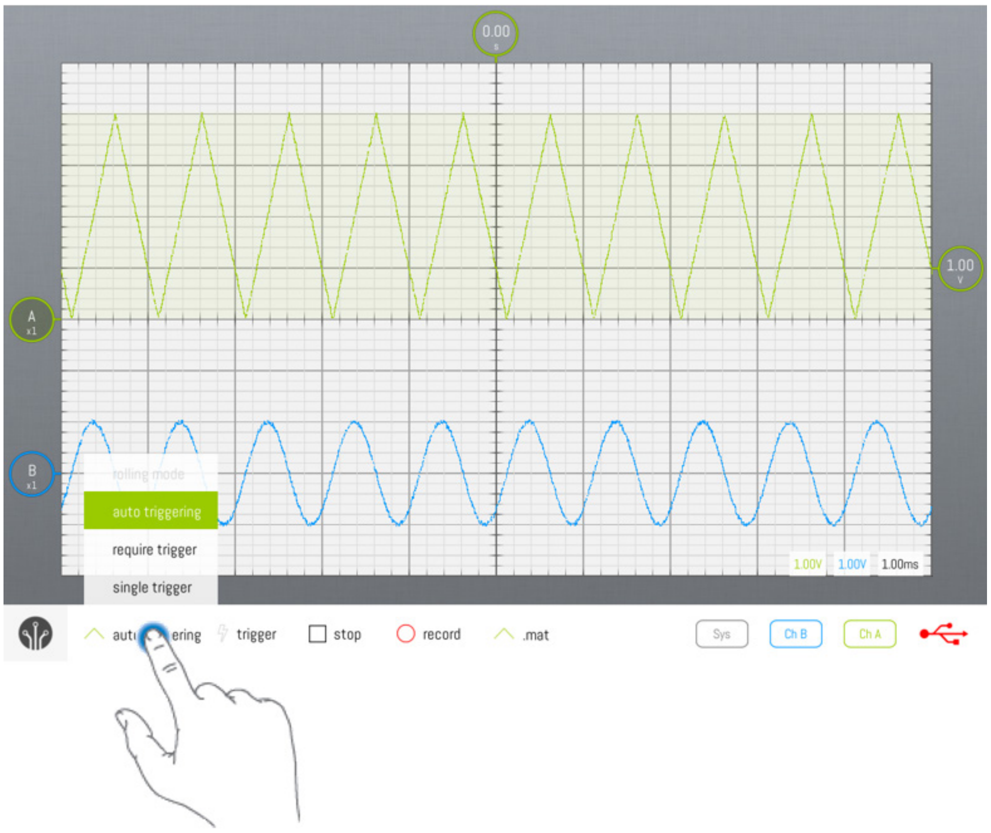


1.1 Trigger indicators

As shown in the image above, the voltage of this crossover point is defined by the Vertical Trigger Indicator on the right side of the screen. Simply drag it up/down to change the trigger voltage. Double-tap the indicator to reset the voltage to 0.00V. Additionally, you can also set the timelocation where you want the crossover point to be rendered. This is indicated by the Horizontal Trigger Indicator at the top of the screen. Simply drag this indicator, or any part of the main viewport, to the left/right to change the trigger location. Double-tap this indicator to reset the trigger position to 0s.

1.2 Trigger modes

By default, the SmartScope will start using Auto triggering mode, and will switch to Rolling mode whenever you zoom the timebase beyond 20ms/div, to visualize slow signals. However, you can freely select the trigger mode of your choice by tapping on the Trigger type drop-up list, as shown below. Notice that the ‘Rolling mode’ is only available when you’re visualizing slow signals (20ms/div and up).



1.2.1 Auto mode

In Auto mode, every time a trigger is detected, the waveform is acquired, transferred to the host and visualized. Whenever the trigger is lost (eg: when the input signal is removed, or when the signal changes so no more crossings with the trigger level occur), the SmartScope sends any acquired waveform to the host for visualization

1.2.2 Normal mode

In Normal mode, every time a trigger is detected, the waveform is acquired, transferred to the host and visualized, much like Auto mode. However, in case no trigger is detected, no new data is transferred nor visualized, keeping the last triggered waveform on the screen.

1.2.3 Single mode

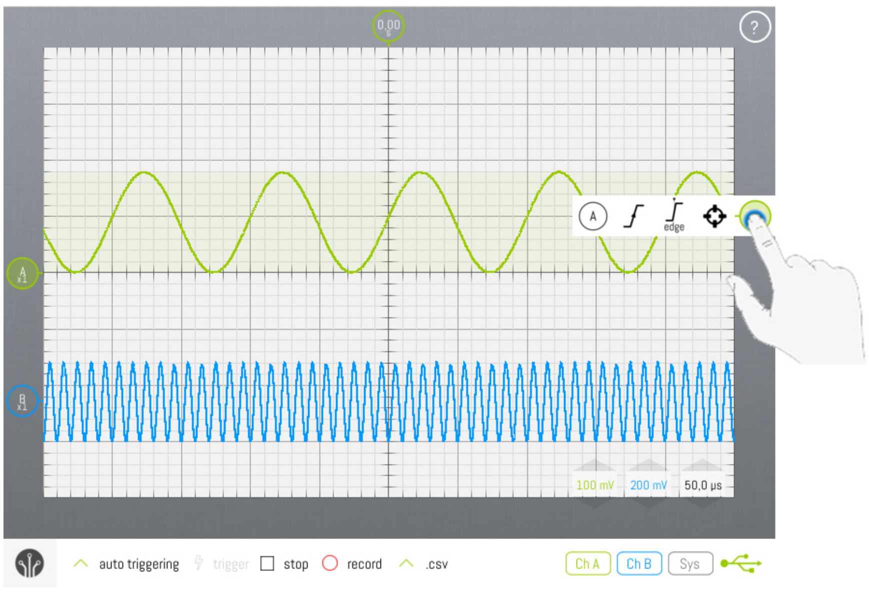
Single mode puts the SmartScope in waiting mode until a trigger is detected. In such case, the waveform is acquired and visualized, after which the SmartScope puts itself in Stopped mode. Single mode is useful in case you want to visualize the very first occurrence of a trigger, or in case you want to capture a unique event and don't want to remove the result in case of some glitch occurs when you remove the probe.

1.2.4 Rolling mode

When you're visualizing slow signals, you end up with timebase settings like 100ms/div. With 10 divisions, this means it takes 1 seconds before an acquitisition can be made, so you only get screen refreshes every second. In such case, it is usually preferred to visualize the data as a continuously incoming stream of data, as you get realtime updates of the signal and you can stop the SmartScope whenever you want. This is what Rolling mode does.

1.3 Trigger menu

The Trigger Menu is opened by tapping the Horizontal Trigger Indicator at the right of the screen:



This will allow you to set (from left to right):

- Which channel to trigger on
- Which edge to trigger on
- The currently selected Trigger Type
- (any optional settings depending on the selected Trigger Type)
- The crosshair which will reset the trigger voltage to 0.00V

1.4 Changing trigger channel

You can specify which channel to trigger on, by opening the Trigger context menu as shown above, and tapping on the leftmost item. This will display a list of channels on which you can trigger. Changing trigger channel can also be done by hitting the T key when a keyboard is available.

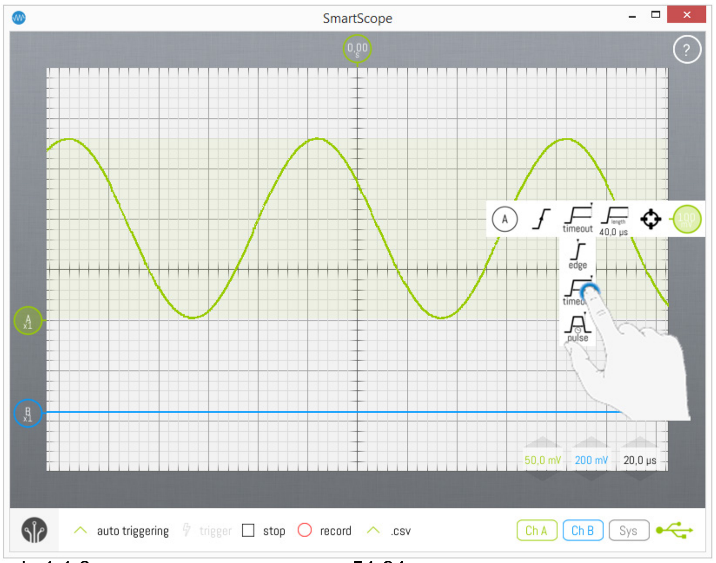
2 Trigger Types

2.1 Edge triggering

The simplest of the three, Edge triggering simply waits for an edge to appear in your signal, and renders that acquisition to your screen. For a Rising edge, this means the position where the voltage transitions from below to above the Trigger voltage. Through the context menu, you can select whether the SmartScope should be waiting for a Rising edge, Falling edge, or whether Any edge will do.

2.2 Timeout triggering

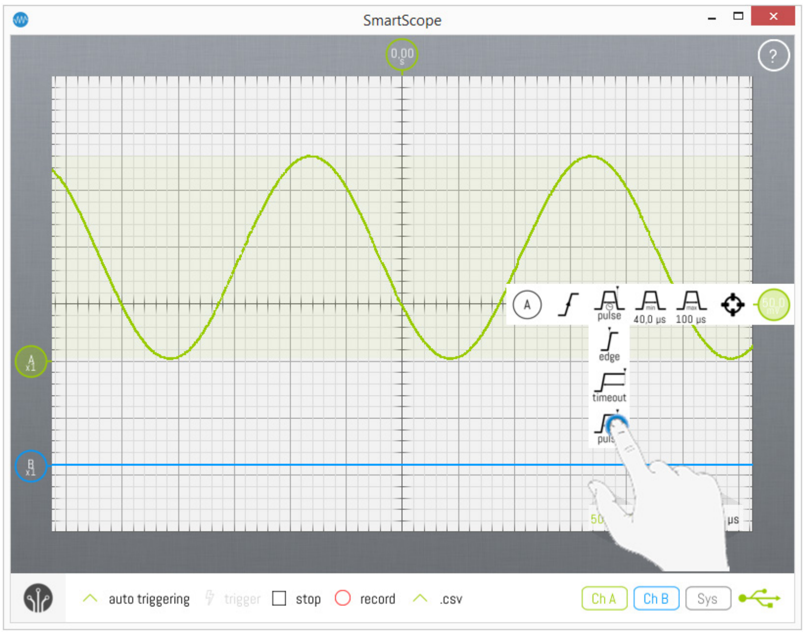
In certain cases, you only want to look at signals which have been above (or below) the Trigger voltage for a certain amount of time. This can be useful to trigger on noise signals, or to detect periods of (in)activity. In order to select Timeout triggering, simply select it from the Trigger menu:



To define the Timeout period, tap on the Length button in the Trigger menu, after which a numpad will show up, allowing you to define your period of choice. Please note the triggering position is NOT the moment of the edge, but rather the moment of the edge + the Timeout period, as can be seen in the Timeout icon. This is demonstrated in the image above, where you can see the positive edge is 40us before the Trigger location.

2.3 Pulse triggering

In other cases, you might be interested in pulses of a certain length. For these cases, the SmartScope supports Pulse triggering. In order to select Pulse triggering, simply select it from the Trigger menu:

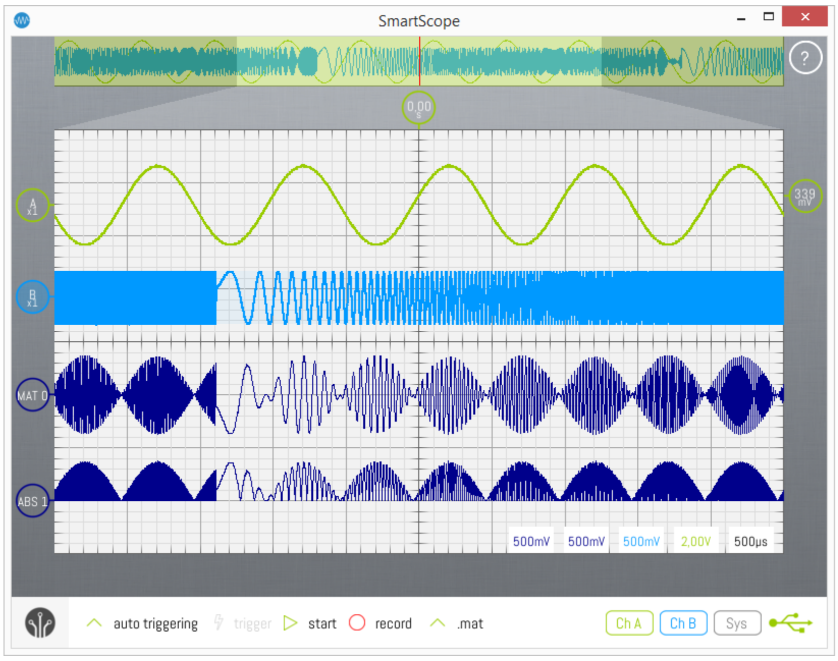


In Pulse triggering mode, you can specify a Minimum and Maximum length. Only pulses within these boundaries will cause the SmartScope to trigger. To set these Minimum and Maximum values, simply tap on them in the Trigger menu after which a numpad will be shown. Please note the triggering position is NOT the moment of the edge, but rather the moment of the end of the pulse, as can be seen in the Pulse icon. This is demonstrated in the image above, where you can see the Trigger location is at the end of the pulse; ie where the voltage drops below the Trigger voltage again.

Using the Operators

The Operator subsystem contains functionality like Math and Average. It allows you to visually compare waves, and perform operations on them. Operators are **stackable**, which means you can use one Operator wave as input to another one. This allows even complex math formulas to be created.

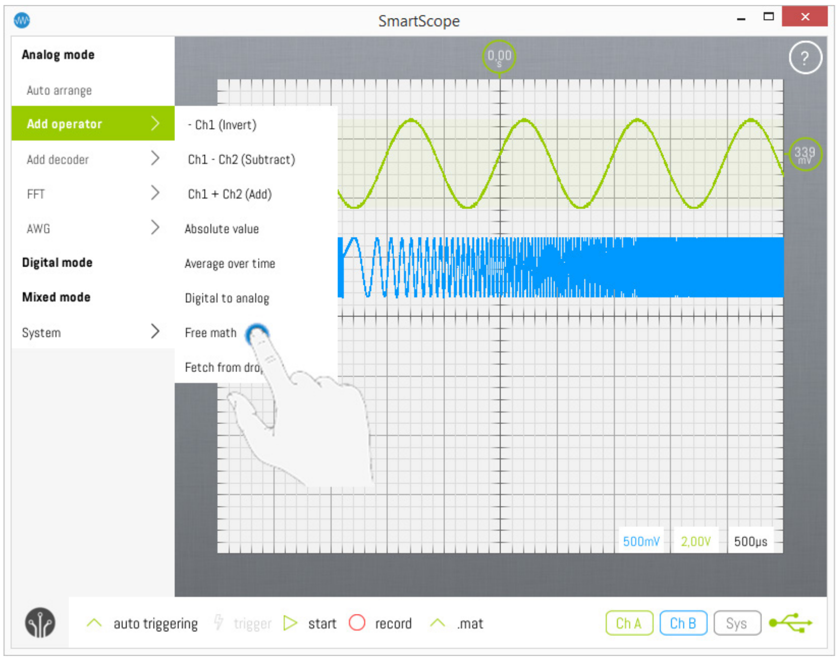
The Operator substystem is completely open and fully extendable. This means you can code your own wave Operators and add it to the SmartScope visualizer. For more info on this topic, see Creating your own Operator.



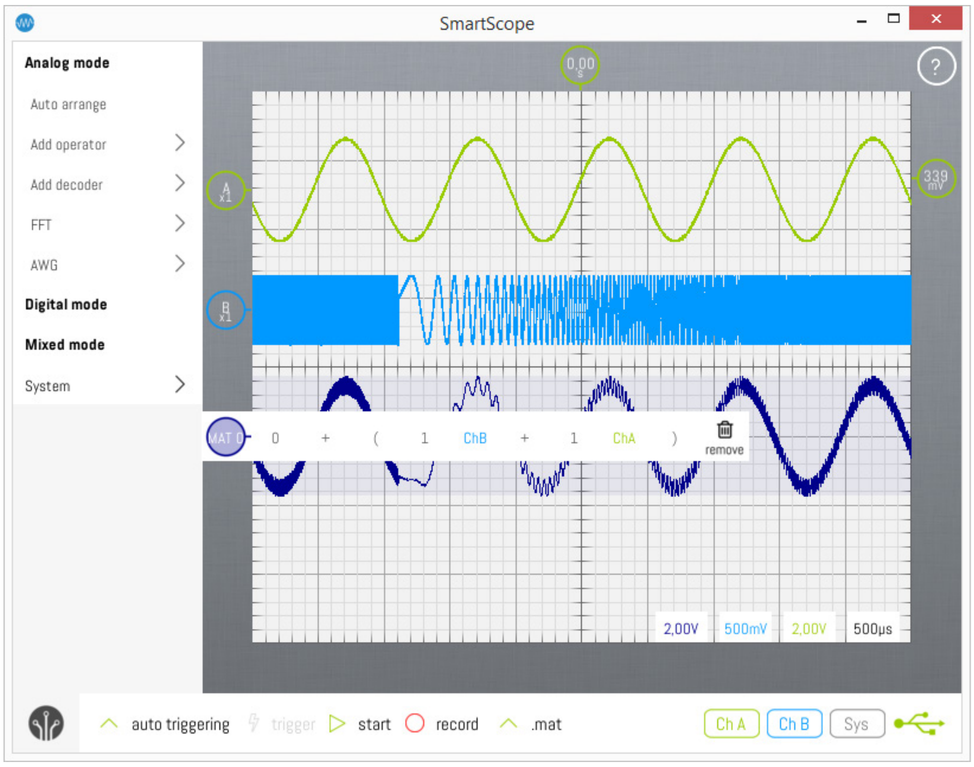
Contents

1 Adding an Operator

Operators are typically used on analog waves, but you can also have operators for digital waves (eg: the digital Invert operator). In order to add an Operator, simply slide open the main menu and hit **Add operator**, which will bring up a list of currently available Operators. (Keep in mind you can easily create your own decoders, see [Creating your own Operator]). Select the Operator you would like to add, as shown in the image below where a *Free math* Operator is being added.



The Operator wave will be added to the main graph, and its context menu will be opened to give you a quick view on its settings (shown in the image below). Finally, just slide it vertically to where you like it to reside.

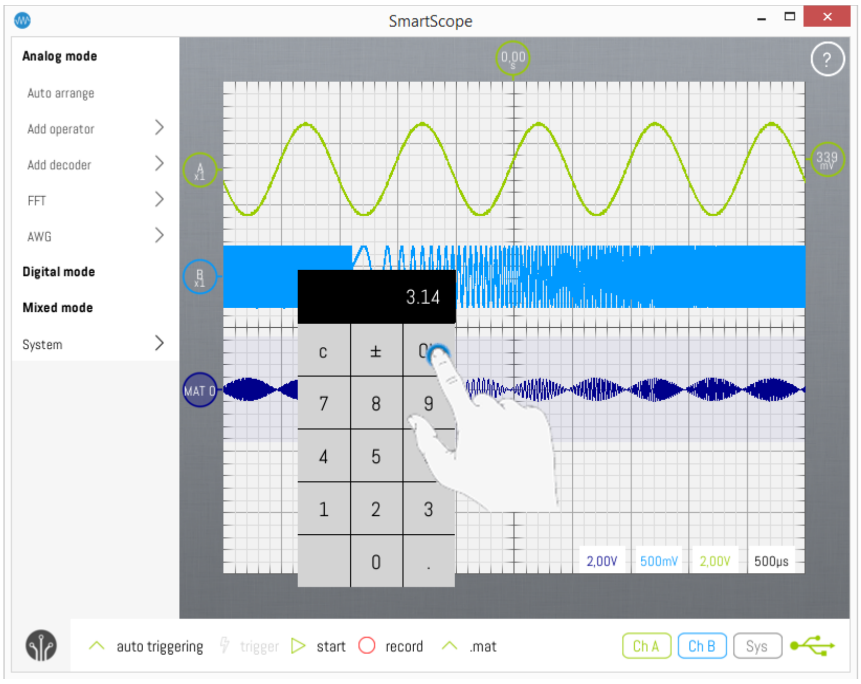


2 Configuring the Operator

All Operators require input waveforms, and usually also some parameters (such as scaling values). In this section you'll see how easy it is to change these Operator settings.

As with all GUI elements of the SmartScope, if you want to configure an Operator, simply tap on the indicator to the left of the Operator. Its context menu will pop up, showing all configurable options (see image above).

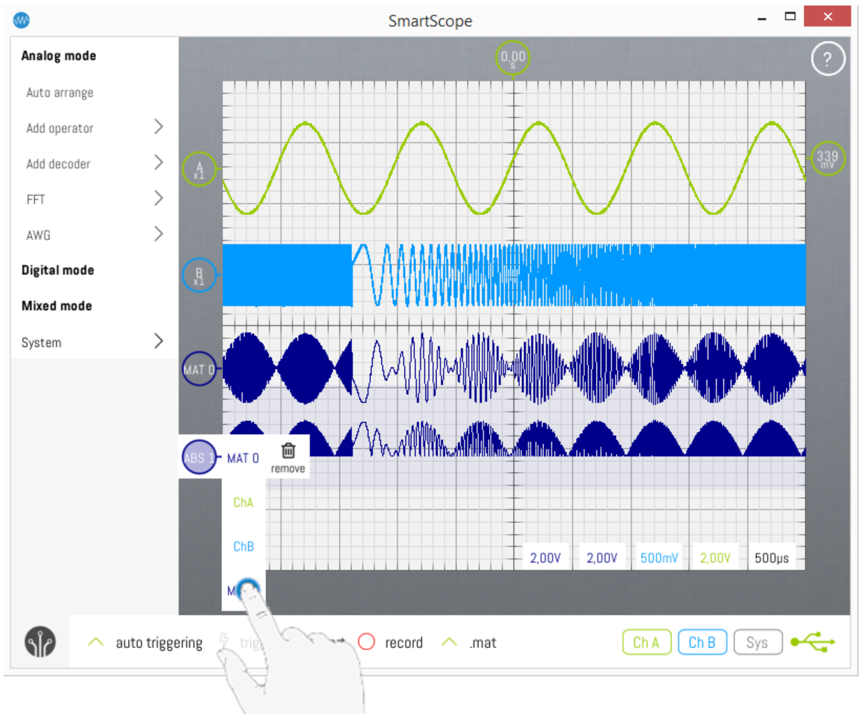
In this specific case of a *Free math* Operator, there are quite some settings: you can select two input waves, and chose whether you want to add, subtract, multiply or divide them. - Simply tap on either wave name, to see a list of possible input wave candidates. - In case you want to change the operator, simply tap the operator to see the list of available operators. - If you want to change a scalar, tap it to see the numpad. The numpad allows you to specify any floating value, as shown in the image below.



Whenever you change any setting of an Operator, the Operator will re-process any available data immediately, also when the acquisition has been stopped. This allows you to acquire a dataset, and get immediately feedback while fine-tuning your Operator settings.

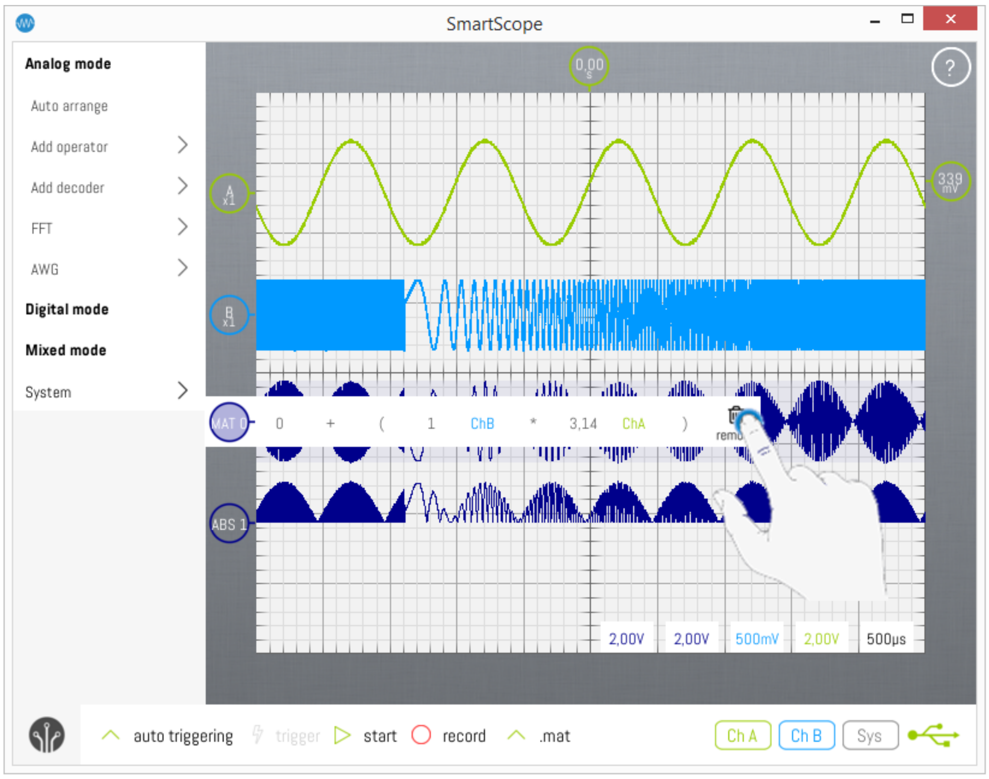
3 Stacking Operators

Any operator can be used as input to another Operator. This allows to create more complicated forumulae. In the following screenshot, the *Absolute* Operator is using the previously created *Free math* Operator as input.



4 Removing an Operator

If you want to remove an Operator, simply tap the Operator indicator on the left, and select the thrashcan. ***Note:*** this will automatically remove any other Operator using the deleted Operator as input channel.



5 Create a custom Operator

See the Creating your own Operator article.

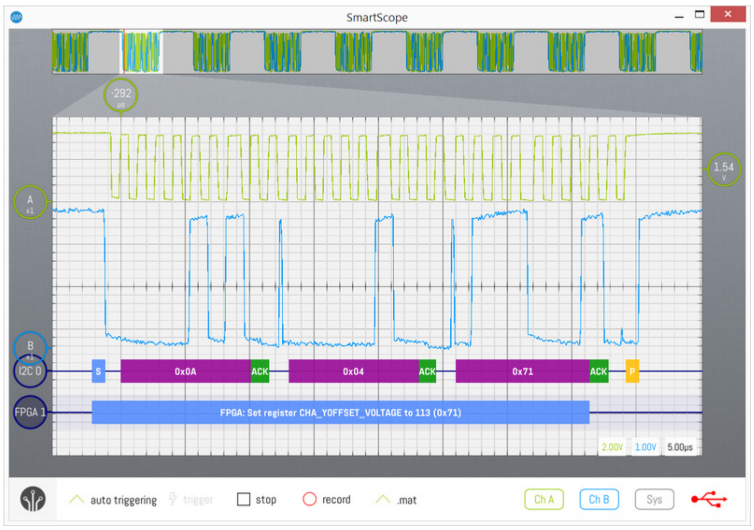
Using the Protocol Decoders

(You can use Protocol Decoders on both the Analog Graph and/or the Logic Analyzer Graph)

While examining the digital communication between 2 chips, did you ever find yourself counting rising edges and writing down 0's and 1's? This is where protocol decoders can save you a LOT of time.

Not only do they convert rising edges into digital data for you; if you select the proper decoder, it will also separate messages and present you byte values. This means you can eg convert a clock and data waveform into I2C byte values, as shown in purple in the image below.

If you want to go one step further, you can even feed these bytes into your own decoder in order to translate them into humanly readable words specific to your application! See the screenshot below, where the output of the I2C decoder is converted into higher-level messages by a custom decoder, as shown in the blue blocks below.

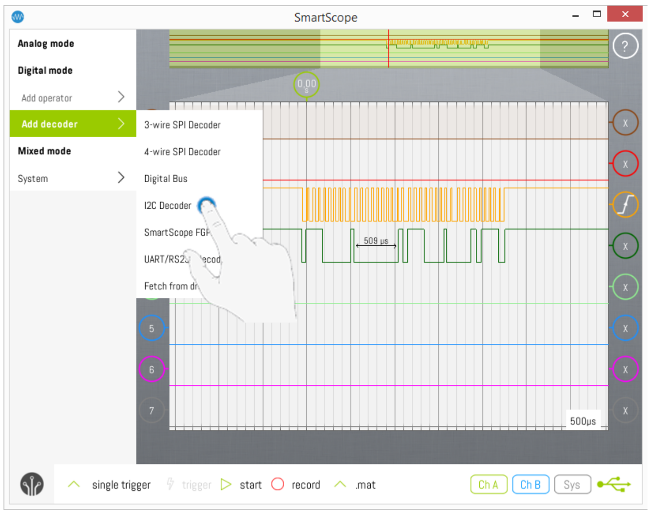


Contents

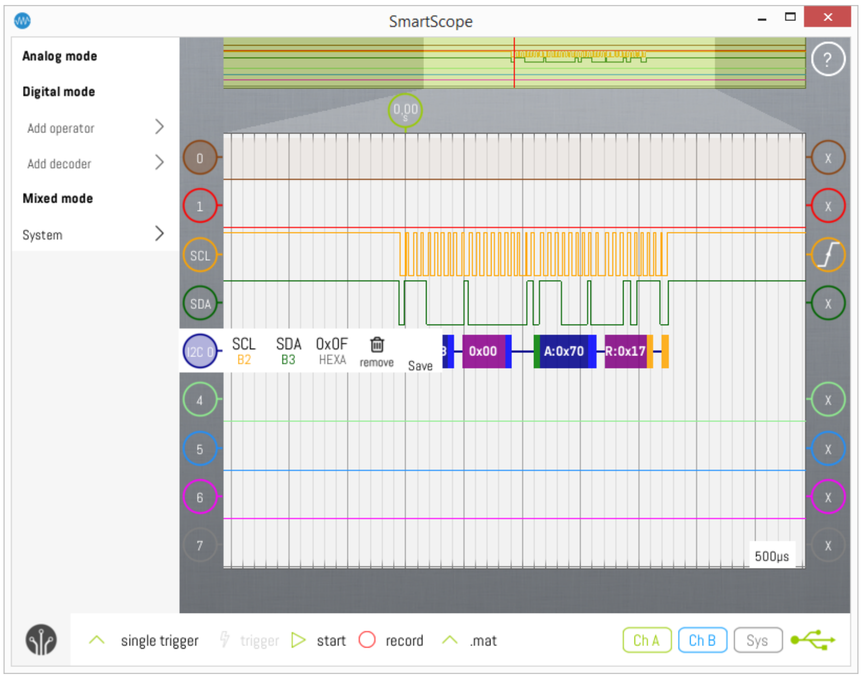
1 Throwing in a decoder

Whenever you feel the need to have a decoder do the bitpicking for you, start by making sure you have the required input signals nicely aligned on your screen. This means you need to make sure the entire communication is visible. - While running, the entire communication should be visible in the Viewport (=main graph) - Preferrably, you'll catch the acquisition using 'Single trigger' mode, as this will allow the SmartScope to fetch the entire acquisition from onboard RAM into the Panorama of the visualizer. Once this has been done, you can get a much more accurate decoding, and you can zoom in to specific portions of interest within a larger communication.

When done, simply slide open the main menu and hit **Add decoder**, which will bring up a list of currently available decoder. (Keep in mind you can easily create your own decoders, see [Creating your own decoder]). Select the decoder you would like to add, as shown in the image below where an I2C decoder is being added.



The decoder wave will be added to the main graph, and it context menu will be opened to give you a quick view on its settings (shown in the image below). The SmartScope software contains a feature which will try to automatically map the available input waves to the currect input of the decoder. Finally, just slide it vertically to where you like it to reside.

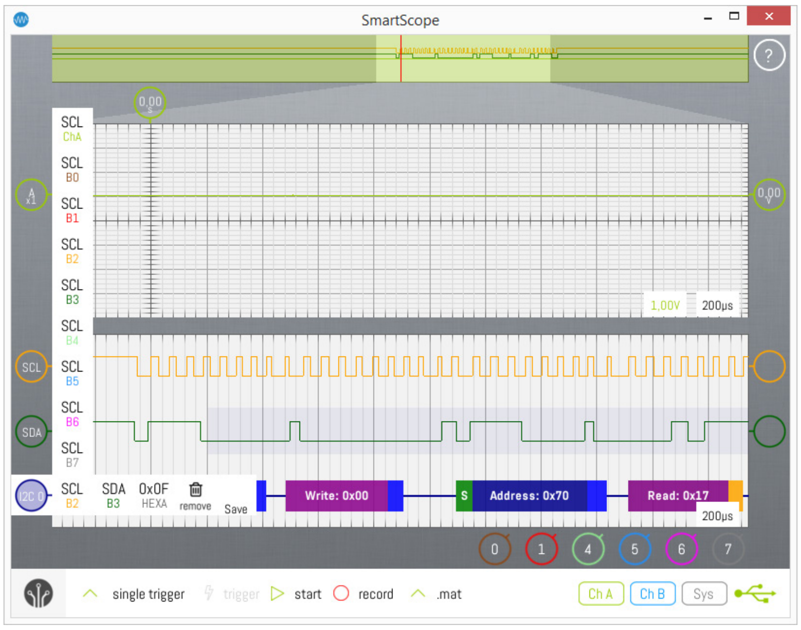


2 Configuring the decoder

All decoders require input waveforms, usually multiple of them. The SmartScope software contains a feature which will try to automatically map the available input waves to the currect input of the decoder. However, in some cases you might want to manually configure the input waves, or other parameters. In this step, you'll let your decoder know which waves to use as which input.

As with all GUI elements of the SmartScope, if you want to configure the decoder, simply tap on the decoder. Its context menu will pop up, showing all configurable options (see image above).

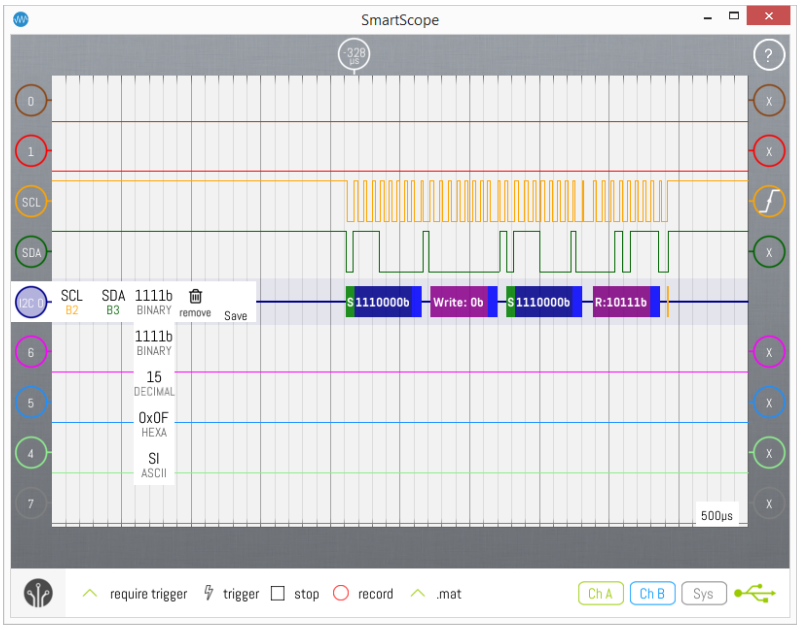
The first entries define which waves are linked to which input. In this specific case of an I2C decoder, there are 2 required inputs: the clock channel SCL and data channel SDA. In case you want to change which wave is being used as input, simply tap on that input, and a list of valid input candidates is shown. In the example below, SCL is tapped, bringing up a list of all possible input wave candidates. For digital inputs, such as SCL in our case, you can choose from all digital and analog waves. This is the reason why the following images was made in Mixed Mode: notice that you can use ChannelA as possible input channel for SCL, even though your I2C Decoder is on your digital graph.



Whenever you change an input channel of a decoder, the decoder will re-process any data immediately, also when the acquisition has been stopped. This allows you to acquire a dataset, and get immediately feedback while fine-tuning your decoder settings.

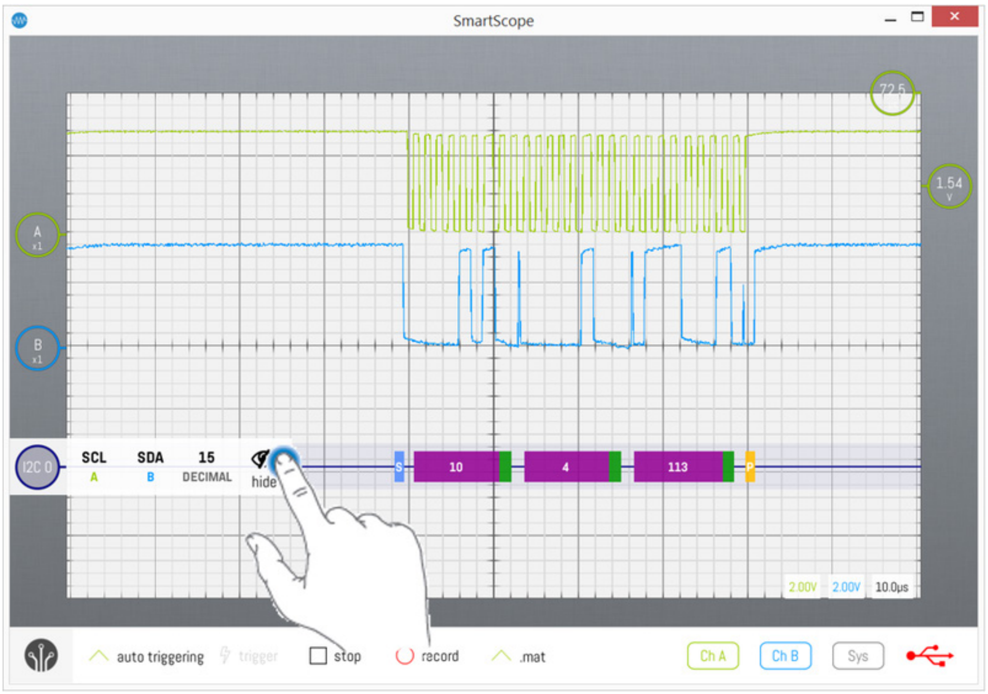
3 Changing the radix of the decoder

By default, decoder output values are shown in hexadecimal. Since not all of us are robots, it might be desirable to change this to decimal values or even other representations. In order to do so, simply tap the decoder indicator on the left, select the radix icon (second to the right), and select the radix of your preference! Currently supported radices include Hex, Decimal, Binary (shown below) and ASCII which will convert the byte value into its ASCII character.



4 Removing a decoder

If you want to switch back to manual bitpicking, simply tap the decoder indicator on the left, and select the thrashcan.



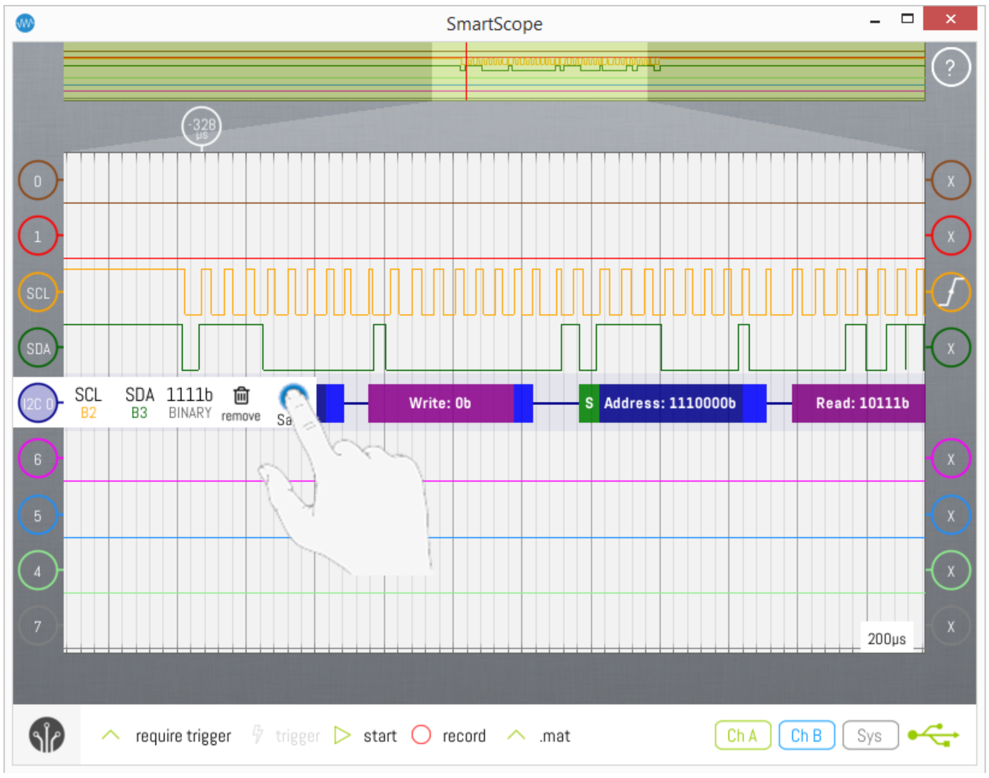
5 Decoding while running or when stopped

While the acquisition is running, the decoding is happening on-the-fly on the data is it is coming in. However, in this mode the decoders only have access to the data shown on the screen. Basically, if you cannot see all edges separately in the Viewport (the main graph), you cannot expect the decoders to decode on-the-fly.

Therefore, if you're not zoomed in enough, correct decoding will not be possible. If this is the case, you can stop the acquisition, after which each and every sample inside the RAM will be transfered to the software, allowing the decoders to do their processing in fine detail.

6 Save all decoded data to file

Visualizing the decoded data is nice, but in some cases you're really interested in the decoded data itself. To do so, open up the Context menu of the Decoder and tap the right-most icon 'Save'. This will store all decoded data in CSV format, including the start- and stop sample of each block, the type of each block and its value.

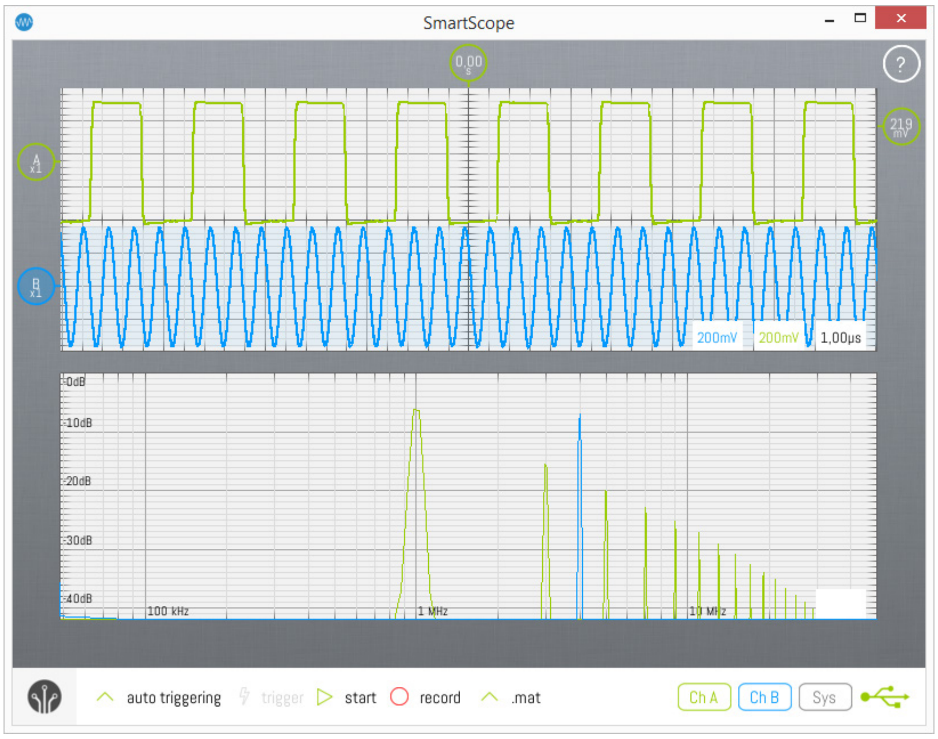


7 Create a custom Protocol decoder

See the Custom Protocol Decoder article.

FFT

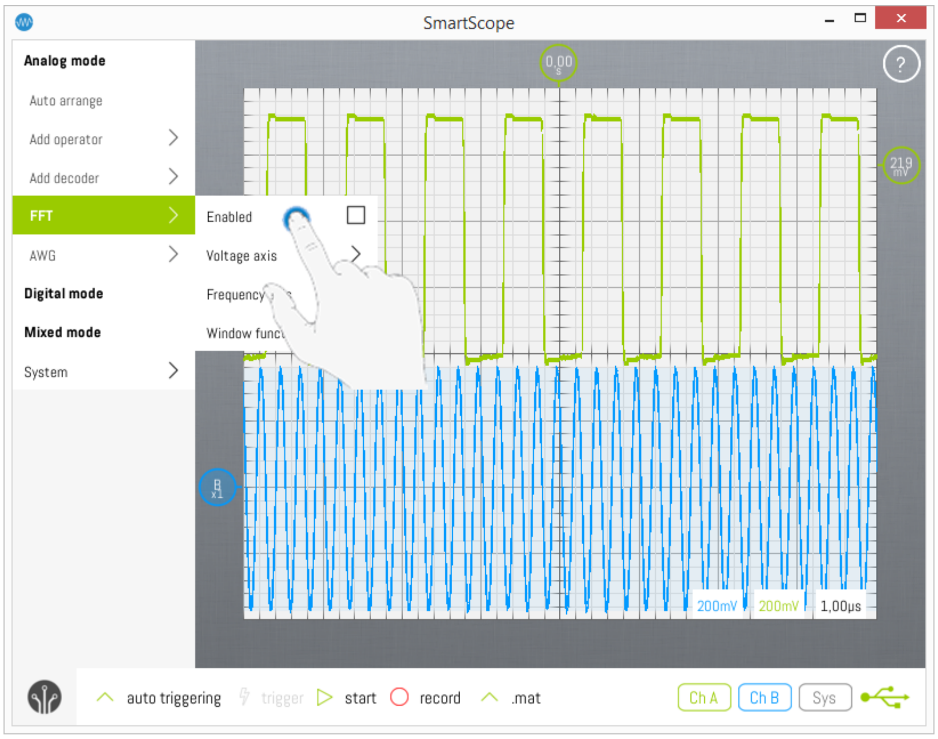
The SmartScope software comes with an integrated spectrum analyzer up to 50MHz.



Contents

1 Enabling FFT

Since FFT is done on the Analog input, first make sure you're in Analog mode after which you can enable the FFT graph by opening the Main menu -> Analog mode -> FFT -> Enable, as shown in the image below.

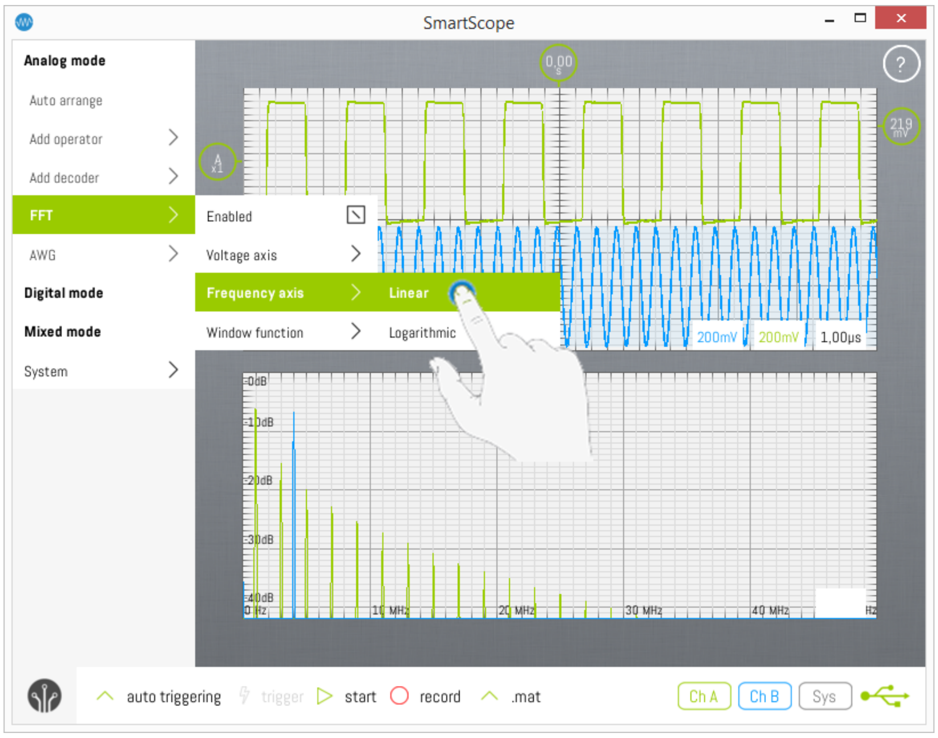


2 Difference between Real-Time and Fine mode

Whenever FFT is enabled and the acquisition is running, a coarse FFT with 2048 bins will be calculated and shown in real-time. In case you want a more accurate FFT, stop the acquisition after which a more fine-grained result (up to 500.000 bins) will be calculated and shown.

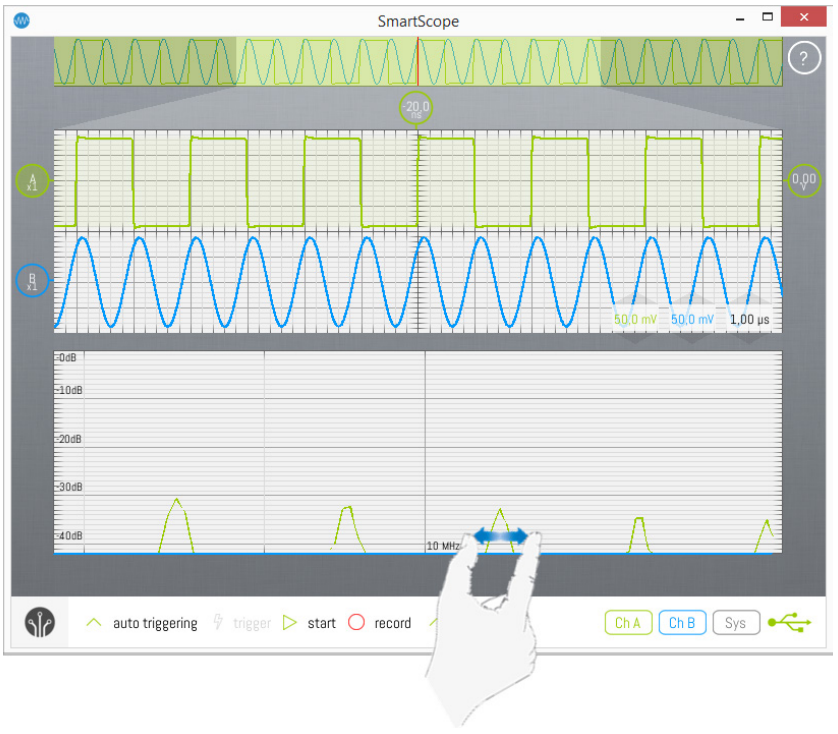
3 Selecting Linear/Logarithmic axes

Especially when working with frequenties, it is often desirable to use logarithmic axes. The SmartScope software allows you to select either linear or logarithmic axes for both the Voltage and Frequency axes individually. The scale of an axis can be changed by going to the **Main menu -> Analog mode -> FFT -> [Axis of your choice] -> [Scale of your choice]**, as shown in the image below:



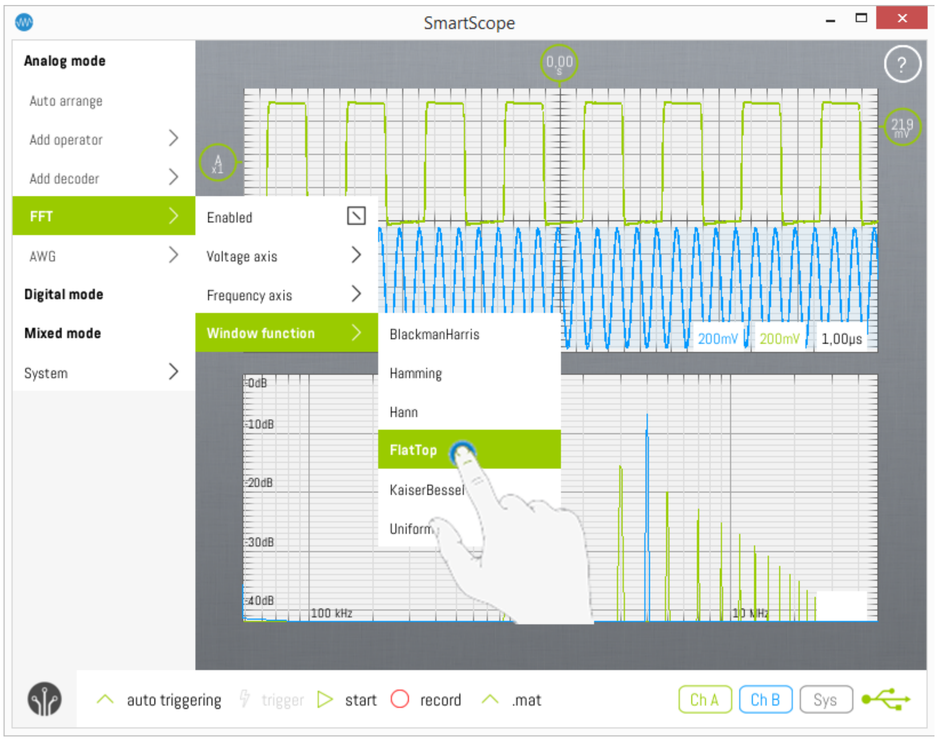
4 Panning/zooming the FFT graph

As of version 0.10, you can now also zoom and pan the frequency axis of the FFT graph, both in Linear and Logarithmic axis. To do so, simply pinch or drag on the graph. You can also zoom using the mousewheel, and pan by dragging the graph with the mouse.



5 Selecting windowing function

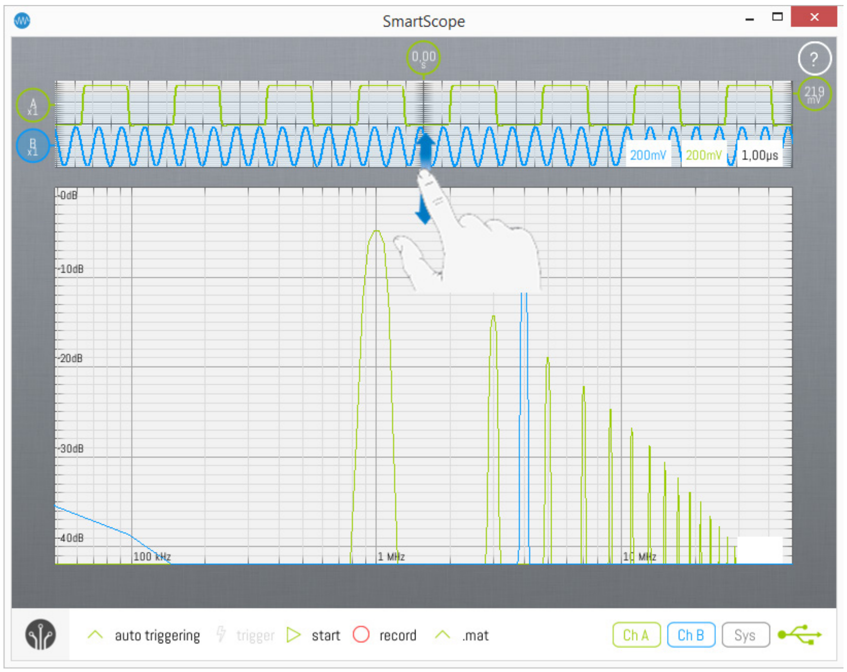
In order to reduce spectral leakage, it is recommended to apply a Windowing function to your signal before taking the FFT. The windowing function can be selected by going to **Main menu -> Analog mode -> FFT -> Window function -> [Window of your choice]**. Select Uniform to turn off windowing.



130

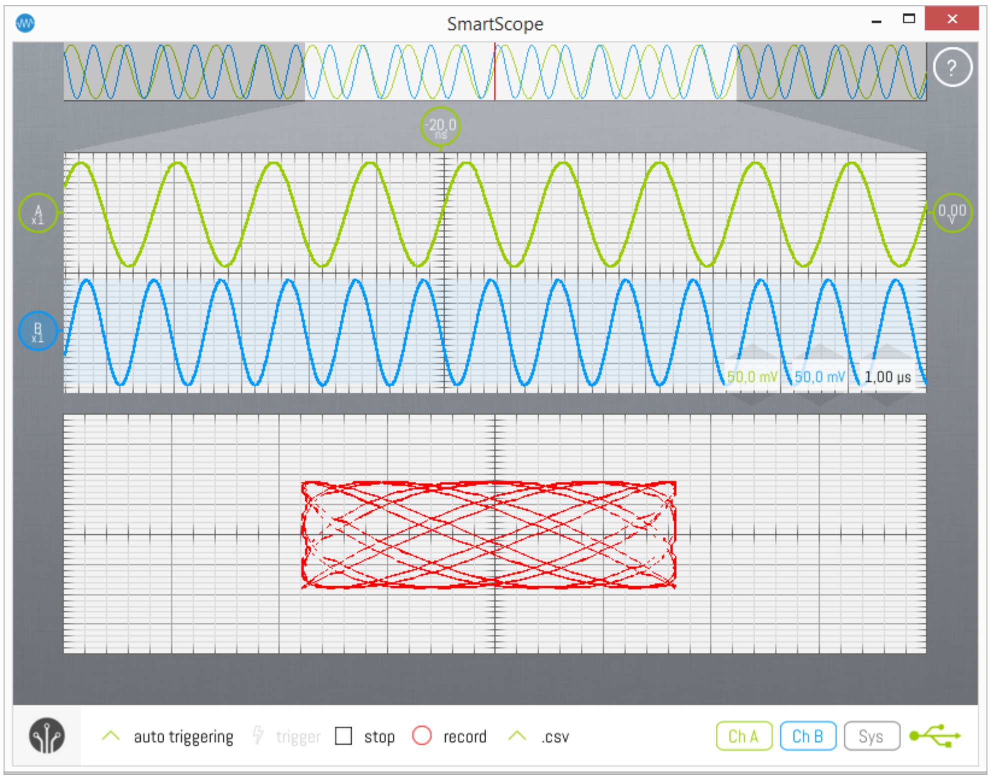
6 Changing size of graphs

In case you want to enlarge the size of either graph, simply drag the area between both graphs up or down as shown in the image below:



XY Mode

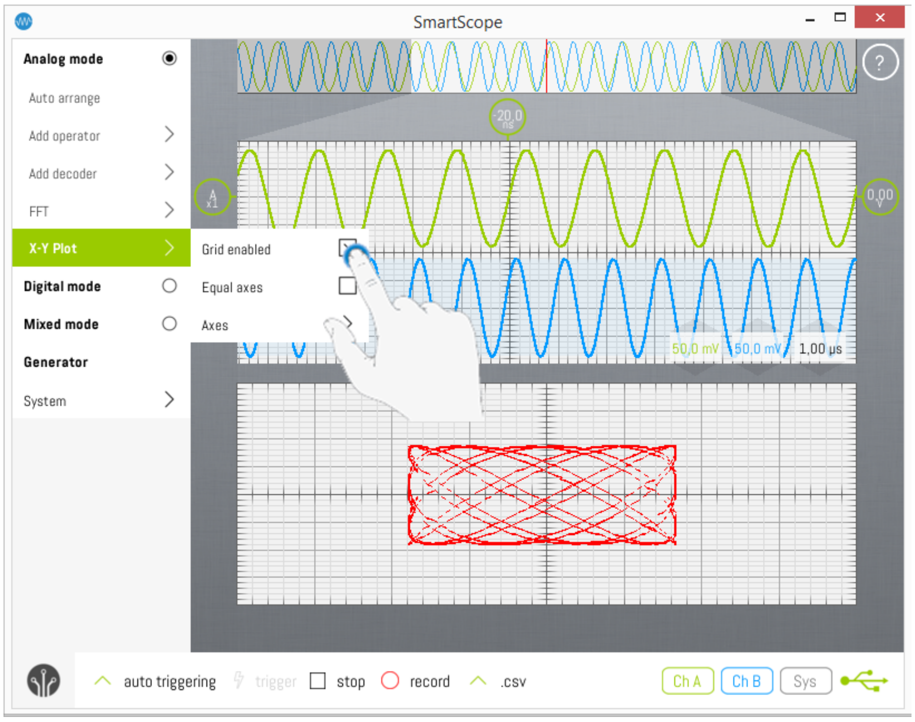
The SmartScope app comes with XY functionality, plotting Channel A versus Channel B (or vice versa)



Contents

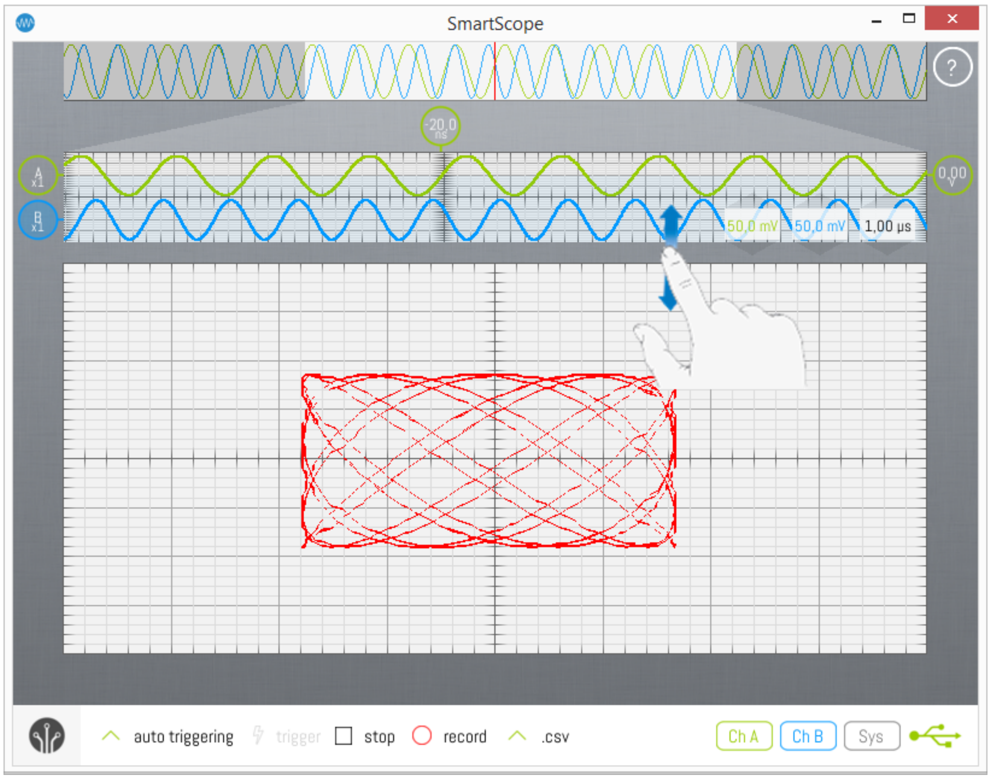
Enabling XY mode

To enable XY mode, open the main menu and select Analog Mode -> XY Plot -> Grid enabled.



Enlarging the XY graph

As with the FFT graph, you can freely select the size of the Analog graph and the XY graph. To do so, grab the gray horizontal border between both graphs, and drag it up or down.

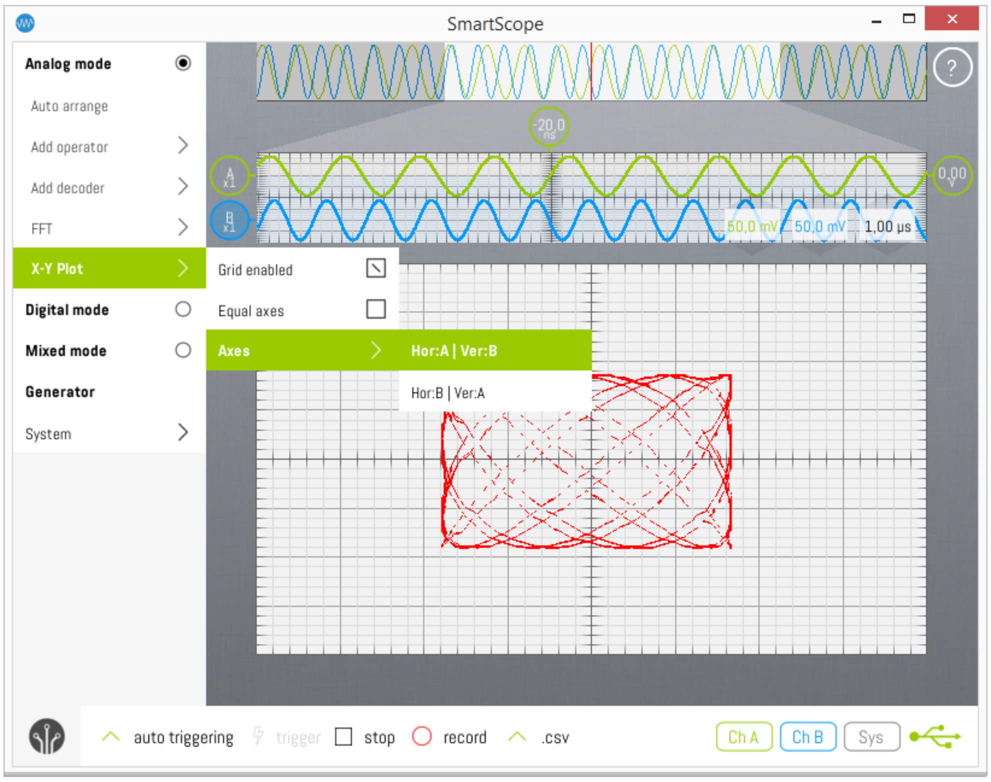


Invert the axis

You can select to either

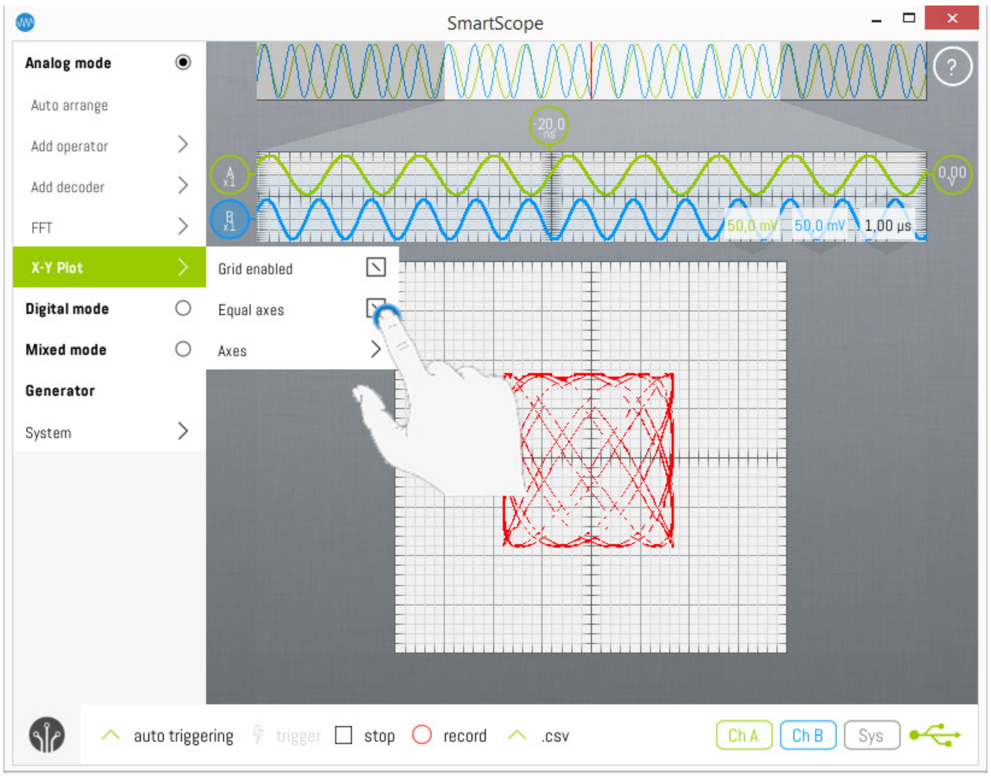
- plot Channel A versus Channel B, or
- plot Channel B versus Channel A

To select your preference, open the main menu and select Analog mode -> XY Plot -> Axes, and select your choice.



Squared XY graph

For some measurements, it is much preferred to have a squared XY plot instead of a rectangular one. To make the graph squared, open the main menu and check the Analog Mode -> XY Plot -> Equal axis checkbox.



This page was last modified on 16 November 2016, at 17:03.

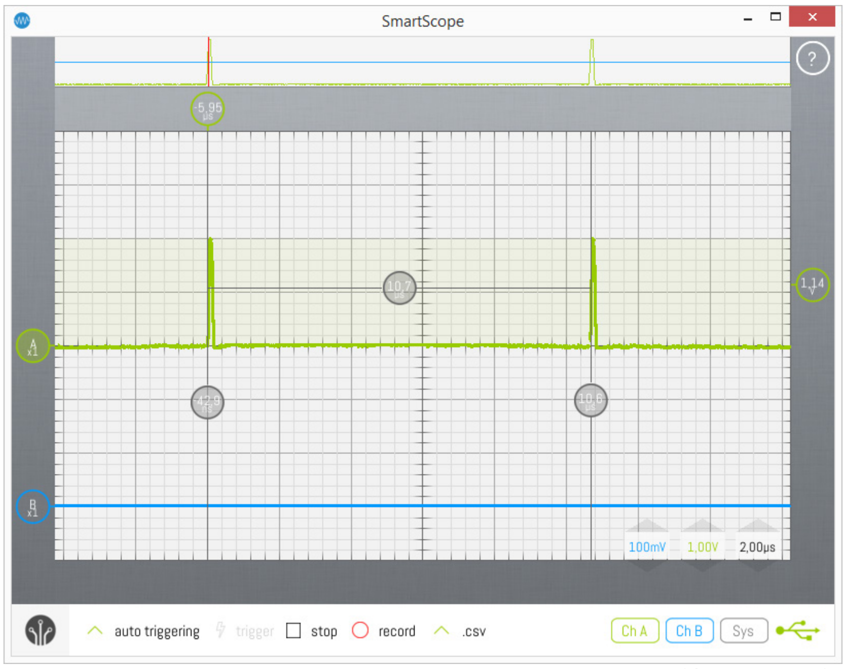
High speed signals - Peak Detect Acquisition - Equivalent Time Sampling

The SmartScope was built for engineers on the road, engineers at home and students. For these use-cases, a digital wave of 10MHz was taken as target, requiring 100MS/s. Since this samplerate is rather low high compared to larger oscilloscopes, the SmartScope software contains a lot of functionality to optimize its performance when visualizing high-speed signals.

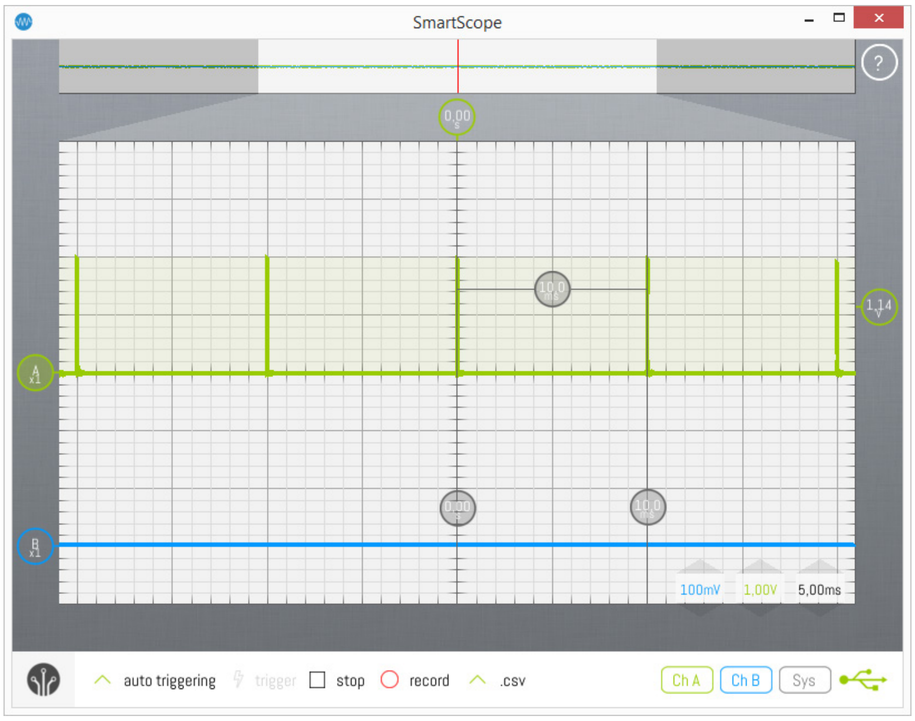
Peak-detect acquisition

What typically happens when zooming out on the timescale, is that the data is being subsampled. By subsampling, there's a high chance tiny spikes would never be seen. The SmartScopes includes peak-detect acquisition, making sure that short pulses are still visible even when you're zoomed out a lot on the timescale.

As an example: on the images below, 10 pulses of 10ns width are generated with 10us between them. This means there's a 1% chance the pulse will be visualized. Since the SmartScope software renders 2048 pixels each acquisition, you'll always see these pulses.



Now when the time between the pulses is enlarged to 10ms, there should be 0.001% chance of seeing the pulses. But because the SmartScope includes peak-detect mode acquisition, **still each and every spike will be shown on the screen**, as you can see in the image below:



(notice that in the Panorama, which does not have peak-detect acquisition implemented, the pulses are indeed not visible)

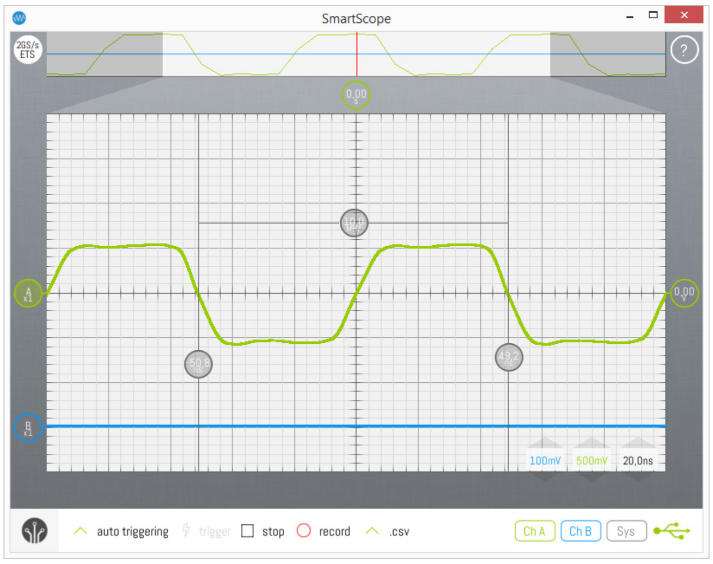
Sin(x)/x triggering

Without sin(x)/x triggering, when zooming in on the timescale approaching 10ns/div, signals could appear jittery as the sampleperiod of the SmartScope is 10ns. The SmartScope software implements reverse sin(x)/x interpolation to predict the trigger-crossing, resulting in a steady triggering even of fast signals. This means **you're ensured each signal will be triggered on the exact crossing of the horizontal and vertical trigger indicators**.

4GS/s Equivalent Time sampling

To top it off, the SmartScope includes functionality which oversamples the signal as you zoom in till 10ns/div. This means that periodic signals will be sampled multiple times, and the visualizer is showing the recombined waveform. As a result, on 10ns/div you'll see the waveform as if it were samples at 4GS/s. This obviously only works for periodic signals, which we're detecting automatically.
You'll see the ETS button at the top-left of the screen when it is activated. Tap it to disable ETS or to activate it again.
Note: in order for ETS to be available, the horizontal trigger position should be within the current timerange of the main graph. In other words: it should not be off-screen.

The image below **shows how a 10MHz digital signal is visualized** when ETS is activated:



(notice once more that the Panorama, which does not have ETS implemented, displays the 10MHz square as it would be displayed without ETS).

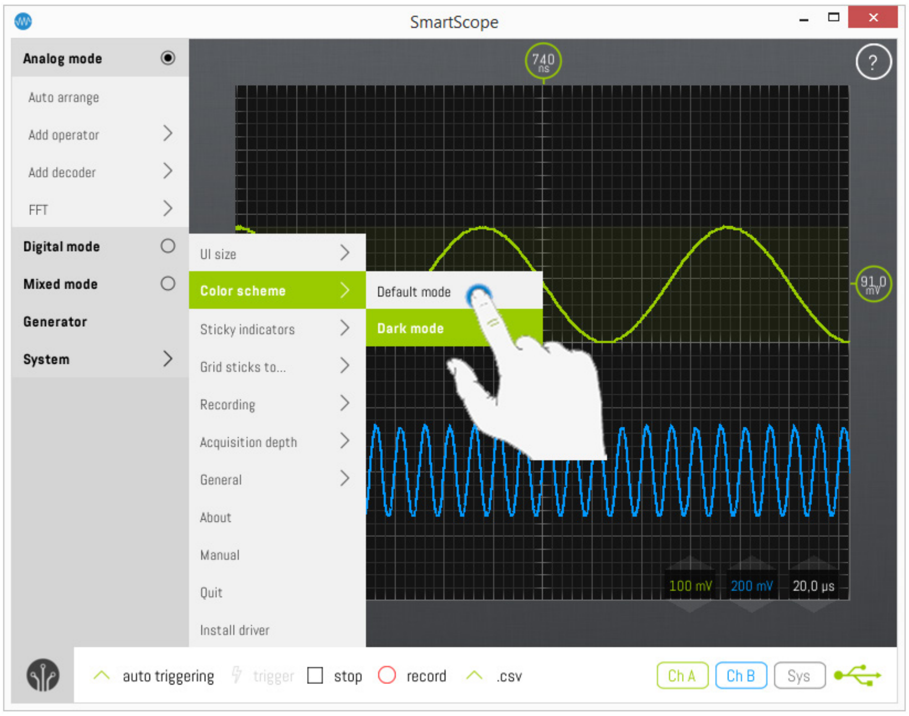
Global/basic functionality

Changing the appearance of the app

The SmartScope software contains a couple of features which allow you to slightly tune its appearance to your liking.

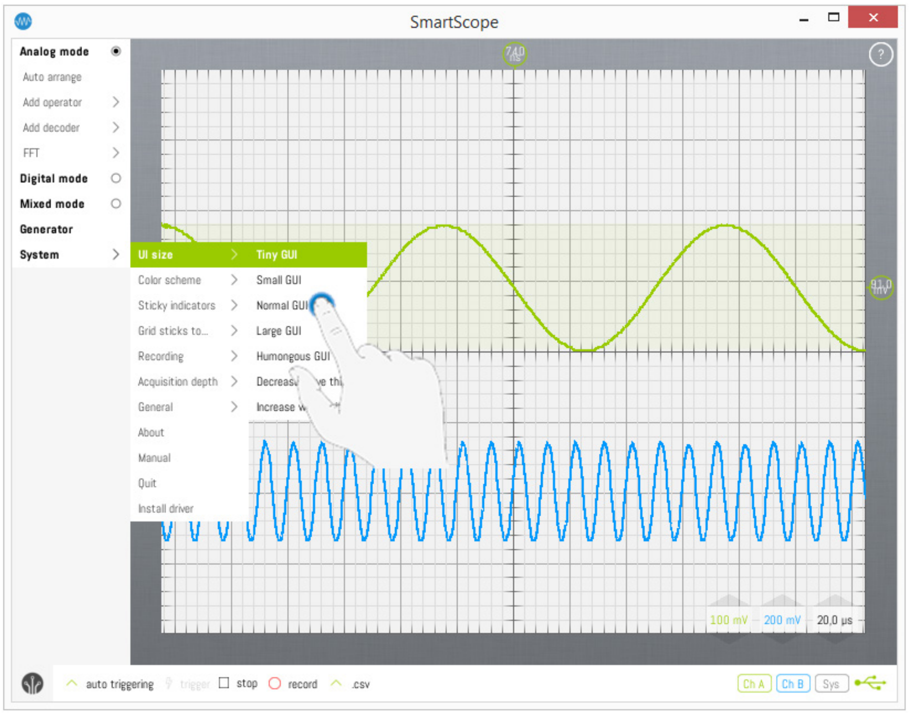
Changing color mode

At this moment, the SmartScope software comes in two color schemes, the bright ‘Default’ mode and the ‘Dark’ mode. You can change between color mode at any moment during operation. To do so, go to Menu -> System -> Color scheme and pick the color mode of your choice.



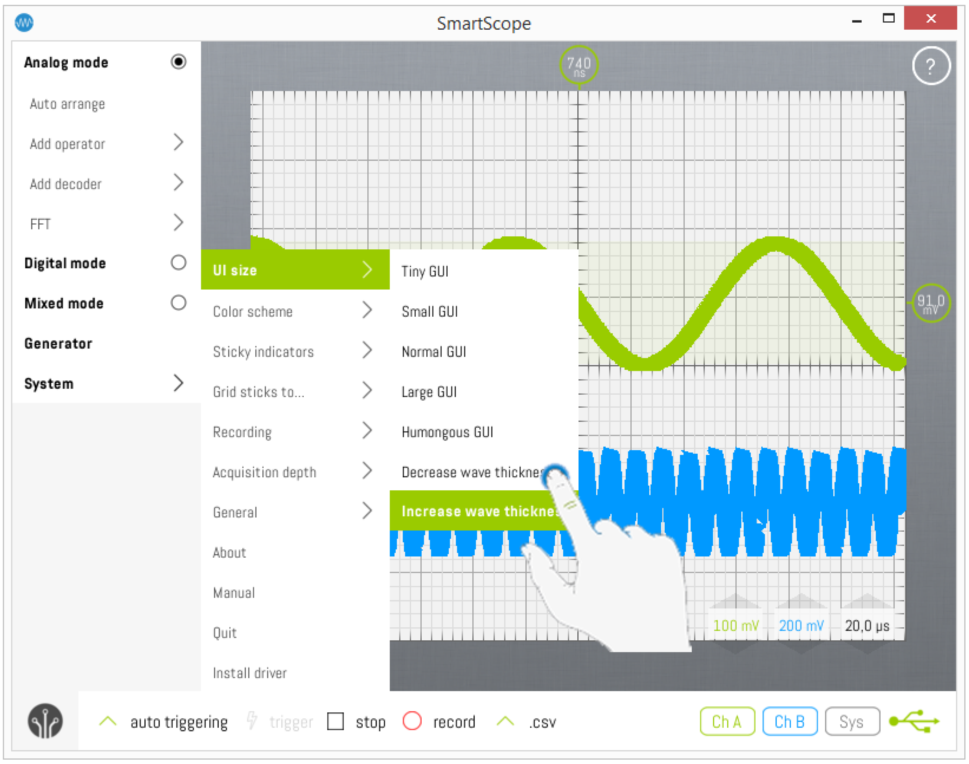
Changing the size of the GUI elements

Determining the size of all interface elements is not so straightforward, because we need to support low resolutions on large PC monitors, as well as ultra-high resolutions on tiny smartphones. We have our algorithms, but in case you would prefer larger/small icons you can adjust your settings by 2 degrees (larger or smaller). Simply go to Menu -> System -> UI Size and try which one you prefer.



Waveform thickness

The default thickness in which your waves are rendered is determined automatically, but you can also set this manually. To do so, go to Menu -> System -> UI Size and select Decrease/Increase wave thickness:



Keyboard shortcuts

Key	Action	CTRL + Action
PgUp	Vertical zoom in on selected wave	
PgDn	Vertical zoom out on selected wave	
Home	Horizontal zoom out on selected wave	Horizontal zoom out on Panorama
End	Horizontal zoom in on selected wave	Horizontal zoom in on Panorama
A	Auto trigger	
S	Single trigger	
D	Require trigger (AKA normal)	
T	Set trigger to selected channel	
Tab	Select next channel	
Shift-Tab	Select previous channel	
F2	Show/hide system measurement box	
F3	Show/hide Channel A measurement box	
F4	Show/hide Channel B measurement box	
Q	Quit	
1	Set digital trigger to '1' for currently selected channel	Same, and select next channel
0	Set digital trigger to '0' for currently selected channel	Same, and select next channel
R	Set digital trigger to RISING for currently selected channel	Same, and select next channel
F	Set digital trigger to FALLING for currently selected channel	Same, and select next channel
X	Set digital trigger to DON'T CARE for currently selected channel	Same, and select next channel
Space bar	Start/Stop acquisition	
P	Toggle Panorama	

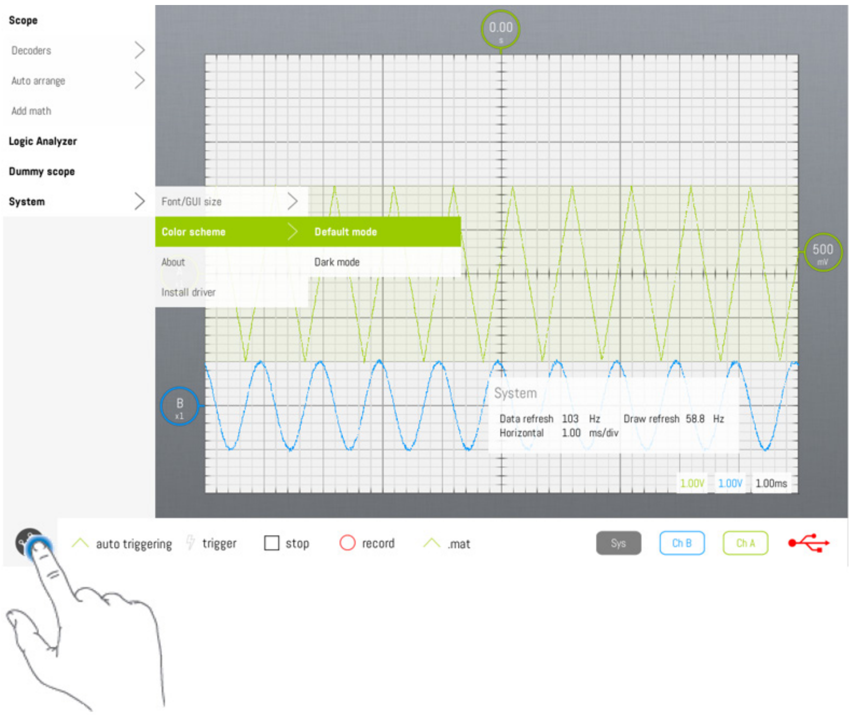
Main menu

Before starting designing the SmartScope app, we wanted to break with the typical oscilloscope interface.

There is no reason to have a dark screen, or a plethora of knobs and buttons. Instead, we wanted to focus on having a clean and simple UI, yet capable of storing all of the complex functionality available in traditional oscilloscopes.

Therefore, **most of the functionality is integrated in each physical element**. For example: if you want to change trigger settings, simply click/tap the trigger indicator on the right of the screen. More general settings have been moved into the menu, which is hidden by default in order to free up as much as possible screen space for the graph.

This **main menu can be opened and closed by tapping on the LabNation logo** in the bottom-left.

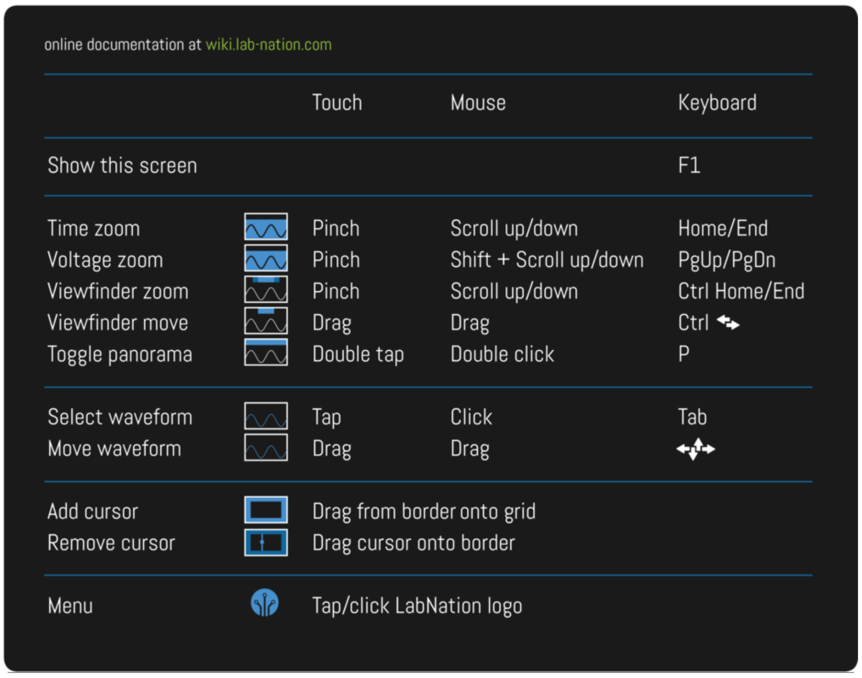


System

Data refresh 103 Hz Draw refresh 58.8 Hz
Horizontal 1.00 ms/div

Cue card

When the SmartScope app is started for the very first time, the Cue Card it shown. Tap on the Cue Card to make it disappear.
The Cue Card contains a quick summary of how to perform the most important operations. You can bring up the cue sheet at any time by pressing the F1 or ? button.
The Cue Card is shown below as fast reference.



online documentation at wiki.lab-nation.com

		Touch	Mouse	Keyboard
Show this screen				F1
Time zoom		Pinch	Scroll up/down	Home/End
Voltage zoom		Pinch	Shift + Scroll up/down	PgUp/PgDn
Viewfinder zoom		Pinch	Scroll up/down	Ctrl Home/End
Viewfinder move		Drag	Drag	Ctrl ◂▸
Toggle panorama		Double tap	Double click	P
Select waveform		Tap	Click	Tab
Move waveform		Drag	Drag	◂▴▾▸
Add cursor		Drag from border onto grid		
Remove cursor		Drag cursor onto border		
Menu		Tap/click LabNation logo		

Left-handed mouse patch

Our software is built on the MonoGame framework, which doesn't natively support left-handed mice. Therefore, our software doesn't support left-handed mice out of the box. However, as discussed in this thread, Rayco from our forum patched the MonoGame dll to work for left-handed mice. You can download the resulting dll here, copy it over the existing dll inside the folder containing the SmartScope executable and you should be fine.

Note: in case this doesn't work, let us know in the forum as we might need to update the dll once in a while.

Customizing/extending the functionality of your SmartScope

Creating your own Operator

The Operator framework has been created from the ground up with extensibility in mind. In case you want to use your own decoder in the SmartScope app, simply follow these steps (details info further down):

- Download the Decoder source package containing the default decoders and operators
- Copy an existing operator file
- Change the metadata, like Name and RequiredInputWaveforms
- Code the logic of your operator in the Process method
- Compile the project
- Copy the resulting .dll file into the same folder as the SmartScope app

And voila, your custom operator will show up nicely in the list of available operators!

Contents

1 Download the Decoder source package

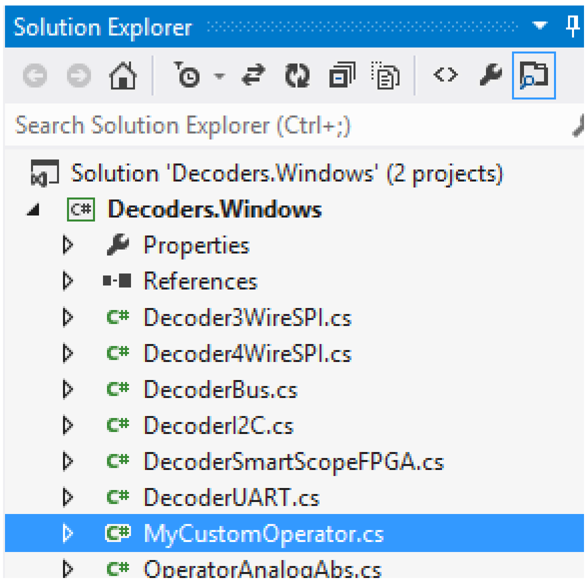
Get the code from github

- Clone it to a local repo if you know how
- Otherwise, simply click the "Download ZIP" button on the right menu to download the full package as a .zip file

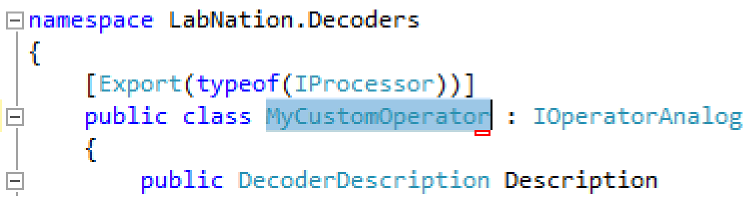
Next, run the Protobuild.exe program to create the solution files for your platform. In case of Windows, this will create (amongst other) the Decoders.Windows.sln file.

2 Start from an existing decoder file

Open up the Decoder project (windows: by double-clicking the Decoders.Windows.sln file).
Next, click on the “OperatorAnalogAbs.cs” file in the Solution Explorer at the top-right of your screen. Hit Ctrl+C and Ctrl+V to duplicate that file. Select the resulting “Copy of OperatorAnalogAbs.cs” file, right-click -> Rename to give the file a meaningful name. In the example below, "MyCustomOperator.cs" was chosen.



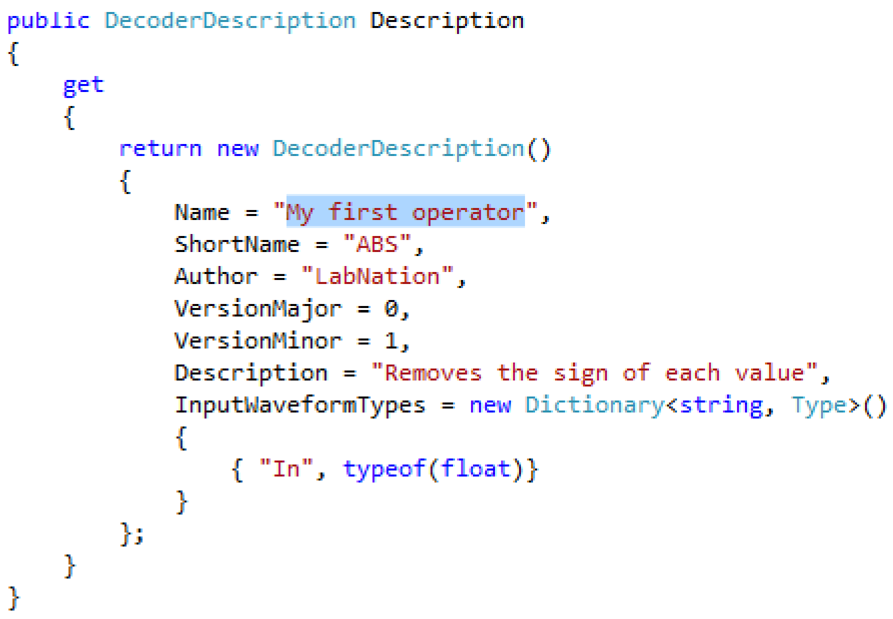
Double-click on the file to open up its contents. Change the original class name OperatorAnalogAbs to a name suiting your filename.



```
namespace LabNation.Decoders
{
    [Export(typeof(IProcessor))]
    public class MyCustomOperator : IOperatorAnalog
    {
        public DecoderDescription Description
```

3 Define the metadata

Time to focus on the code. Just underneath the class name, you'll notice the Description which contains a couple of properties which define everything about your operator the SmartScope app needs to know, except the processing itself. Change all of them to suite your new operator's needs:



```
public DecoderDescription Description
{
    get
    {
        return new DecoderDescription()
        {
            Name = "My first operator",
            ShortName = "ABS",
            Author = "LabNation",
            VersionMajor = 0,
            VersionMinor = 1,
            Description = "Removes the sign of each value",
            InputWaveformTypes = new Dictionary<string, Type>()
            {
                { "In", typeof(float)}
            }
        };
    }
}
```

- **Name:** This is how your operator will be presented by the SmartScope app in the list of available operators.
- **Abbreviation:** Max 4 characters describing your operator. These will be printed inside the indicator of your operator wave at the very left of the SmartScope app.
- **Author:** Your name. Not used at the moment, sorry about that.
- **VersionMajor & VersionMinor:** In case multiple operators with same name are detected, this field allow the SmartScope app to detect which instance it the most recent one.
- **InputWaveformTypes:** Here you define how many input waveforms your operator requires, and which type of data they need to contain. This is done by using a Dictionary, linking the name of the wave to the type of required contents.

Optional:

- *Parameters:* Allows the user to specify additional paramteres, such as ActiveLow/ActiveHigh, or values such as Baud rates etc. See the OperatorAnalogMath for an example on this.
- *ContextMenuOrder:* Allows you to specify the order in which the InputWaves and Parameters will be presented in the Context menu of the SmartScope app. See the OperatorAnalogMath for an example on this.

4 Code your operation

With all preparations done, it's time to focus on the fun stuff. Head to the Process method, which has 3 arguments:

- **inputWaveforms:** this is a list of arrays, containing all waveforms you specified above in the RequiredInputWaveforms property.
- **parameters:** this is a dictionary containing all paramters you specified above in the RequiredInputWaveforms property.
- **samplePeriod:** in case your operator requires absolute timestamps, this argument is for you. By multiplying this value by the indices of your input arrays, you can find the exact time between them. In case your decoder would require 2 digital waveform and 1 waveform containing voltages (floats), you should have this:

NOTE: if you specify bool, the SmartScope app will pick up all logic channels and analog channels (the app will convert them to binary values for you). If you specify float, the SmartScope app will pick up analog channels only.

Now you'll notice the inputWaveforms is a Dictionary, providing a typeless Array for each input you require. Before you can use the actual values inside the Arrays, you'll want to cast them to Arrays containing the type your defined earlier.

4.1 Operator output

Now you can go ahead and implement your operator. The output of your decoder must be an array of floating values, having the same lenght as the input waveforms. These floating values will be rendered straight to the screen of the SmartScope app.

5 Compile the project

Select Build -> Build solution (F6), solve any bugs and iterate until all bugs have been squashed.

6 Move the .dll file to the right folder

Once compilation is finished, find the resulting .dll file. You can find the location where the .dll is being generated by right-clicking on your project (Decoders in the screenshot at the top of this page), selecting Properties, selecting Build in the menu on the left and then checking the Output path on the bottom of the page presented.

Copy or move that .dll file (Decoders.dll in case of this example) to the following folder:

Platform	**Path**
Mac	/Users/<username>/LabNation/Plugins
Linux	~/LabNation/Plugins
Windows	<My Documents>/LabNation/Plugins
Android	<sd-card>/LabNation/Plugins
iOS	See section below regarding DropBox

Now when you restart the SmartScope app, it should automagically pick up your new operator and list it as available operator!

6.1 Using DropBox to transfer the dll to your tablet/phone

For all platforms, you can access a DLL file over DropBox. To do so, in the app go to Menu -> Add decoder -> Fetch from dropbox. If never done before, this will authenticate to DropBox and create all folders required. Next, on your PC you can save the DLL file to \Dropbox\Apps\LabNation SmartScope\Plugins. All DLL files you place here can now be accessed from all your devices over dropbox!

7 Debugging your operator in real-time

Even though your project passes compilation successfully, there are many reasons why your operator might not produce the desired result, or even cause the SmartScope throw an error message. The best way to figure out why, is to put a breakpoint in your code so you can step through your code while live data is being fed in. In order to do so, simply put a breakpoint at the first line of code of your Process method. Start up the SmartScope app, and add your operator.
Now usually you would expect the code to break at your breakpoint, but not in this case as the SmartScope app is using your compiled .dll, and not the code source you're looking at. But no worries, we just need to let Visual Studio know that your source is exactly the same as the .dll which is currently being executed. To do so, select Debug -> Attach to process. Select SmartScope.exe from the list, and hit Attach. You should see that the SmartScope app is being halted, and your breakpoint is active now. Step through your code using F11 and F10 as you would debug any other program!

Creating your own Protocol Decoder

The decoder framework has been created from the ground up with extensibility in mind. In case you want to use your own decoder in the SmartScope app, simply follow these steps (details info further down):

- Install and learn to use Microsoft Visual Studio
- Download the Decoder source package containing the sample decoder project
- Copy an existing decoder file
- Change the metadata, like Name and RequiredInputWaveforms
- Code the logic of your decoder in the Process method
- Compile the project
- Copy the resulting .dll file into the same folder as the SmartScope app

And voila, your custom decoder will show up nicely in the list of available processors!

Contents

1 Download the Decoder source package

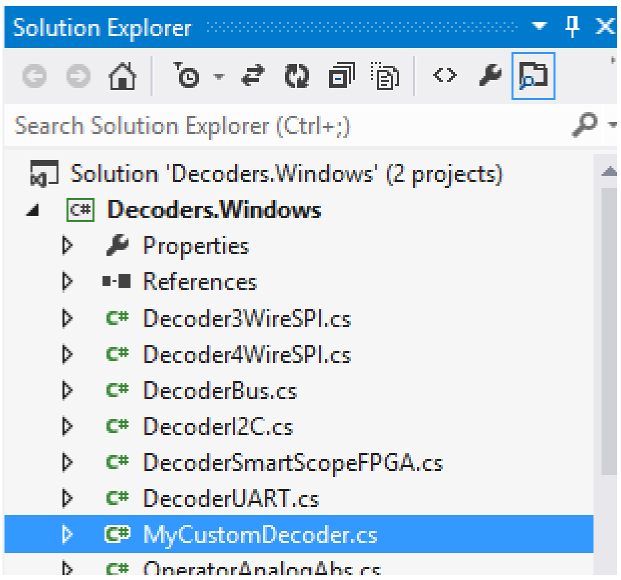
Get the code from github - Some elementary instructions are available there as well

- Clone it to a local repo if you know how
- Otherwise, simply click the "Download ZIP" button on the right menu to download the full package as a .zip file

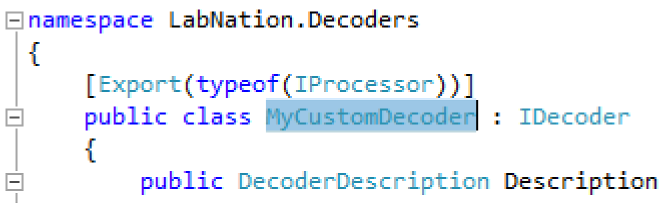
Next, run the Protobuild.exe program to create the solution files for your platform. In case of Windows, this will create (amongst other) the Decoders.Windows.sln file.

2 Start from an existing decoder file

Open up the Decoder project (windows: by double-clicking the Decoders.Windows.sln file).
Next, click on the “DecoderI2C.cs” file in the Solution Explorer at the top-right of your screen. Hit Ctrl+C and Ctrl+V to duplicate that file. Select the resulting “DecoderI2C - Copy.cs” file, right-click -> Rename to give the file a meaningful name. In the example below, "MyCustomDecoder.cs" was chosen.



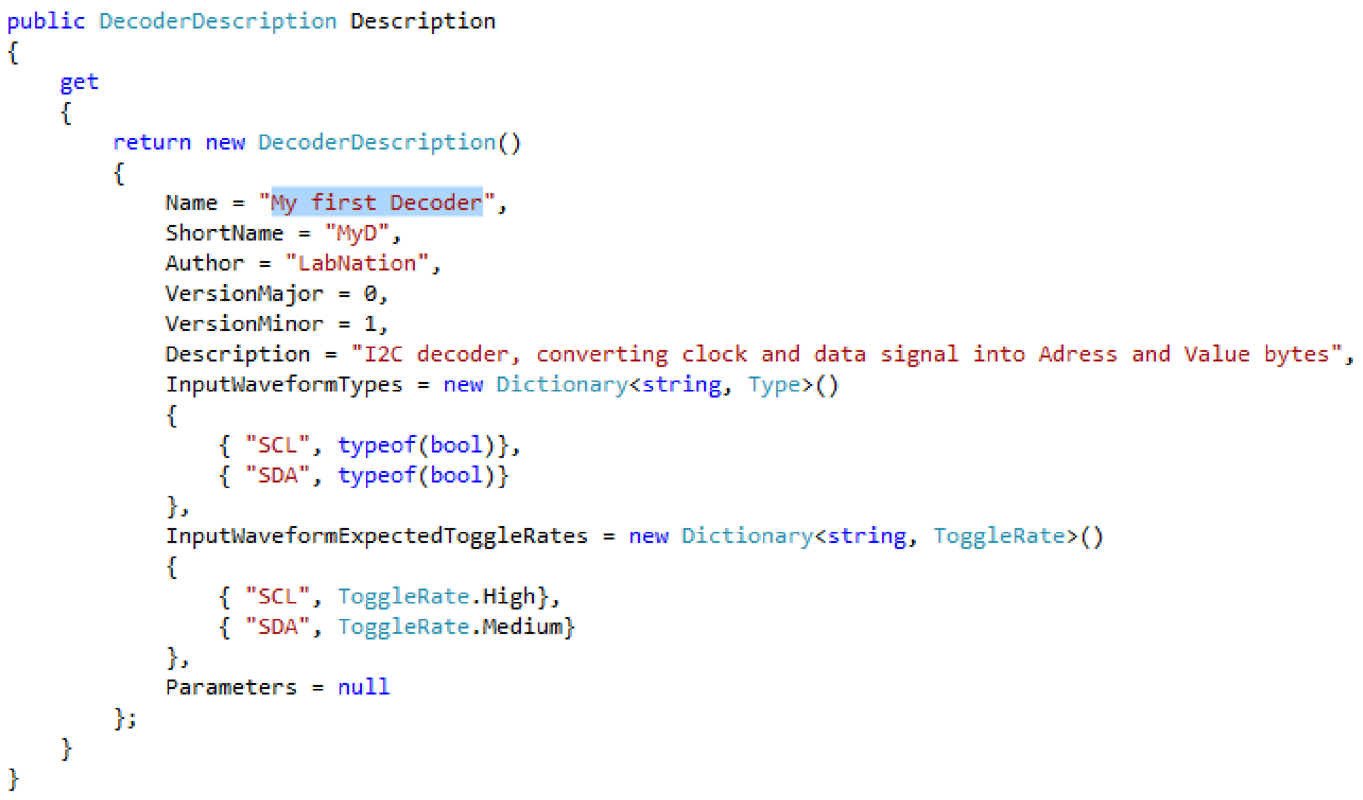
Double-click on the file to open up its contents. Change the original class name DecoderI2C to a name suiting your filename.



```
namespace LabNation.Decoders
{
    [Export(typeof(IProcessor))]
    public class MyCustomDecoder : IDecoder
    {
        public DecoderDescription Description
```

3 Define the metadata

Time to focus on the code. Just underneath the class name, you’ll notice the Description which contains a couple of properties which define everything about your decoder the SmartScope app needs to know, except the decoding itself. Change all of them to suite your new decoder’s needs:



```
public DecoderDescription Description
{
    get
    {
        return new DecoderDescription()
        {
            Name = "My first Decoder",
            ShortName = "MyD",
            Author = "LabNation",
            VersionMajor = 0,
            VersionMinor = 1,
            Description = "I2C decoder, converting clock and data signal into Adress and Value bytes",
            InputWaveformTypes = new Dictionary<string, Type>()
            {
                { "SCL", typeof(bool)},
                { "SDA", typeof(bool)}
            },
            InputWaveformExpectedToggleRates = new Dictionary<string, ToggleRate>()
            {
                { "SCL", ToggleRate.High},
                { "SDA", ToggleRate.Medium}
            },
            Parameters = null
        };
    }
}
```

- **Name:** This is how your decoder will be presented by the SmartScope app in the list of available decoders.
- **Abbreviation:** Max 4 characters describing your decoder. These will be printed inside the indicator of your decoder wave at the very left of the SmartScope app.
- **Author:** Your name. Not used at the moment, sorry about that.
- **VersionMajor & VersionMinor:** In case multiple decoders with same name are detected, these field allow the SmartScope app to detect which instance it the most recent one.
- **InputWaveformTypes:** Here you define how many input waveforms your decoder requires, and which type of data they need to contain. This is done by using a Dictionary, linking the name of the wave to the type of required contents.

Optional:

- *InputWaveformExpectedToggleRates:* The SmartScope app contains functionality which allows to automatically map the correct input signal to the correct input of your Decoder. To allow this, you only need to specify which input sources you expect to toggle the most. This is done by a dictionary, specifying the expected ToggleRate for each input waveform.
- *Parameters:* Allows the user to specify additional paramteres, such as ActiveLow/ActiveHigh, or values such as Baud rates etc. See the DecoderUART for an example on this.

- *ContextMenuOrder:* Allows you to specify the order in which the InputWaves and Parameters will be presented in the Context menu of the SmartScope app. See the OperatorAnalogMath for an example on this.

4 Code your logic

With all preparations done, it's time to focus on the fun stuff. Head to the Process method, which has 3 arguments:

- **inputWaveforms:** this is a list of arrays, containing all waveforms you specified above in the RequiredInputWaveforms property.
- **parameters:** this is a dictionary containing all paramters you specified above in the RequiredInputWaveforms property.
- **samplePeriod:** in case your decoder requires absolute timestamps, this argument is for you. By multiplying this value by the indices of your input arrays, you can find the exact time between them. In case your decoder would require 2 digital waveform and 1 waveform containing voltages (floats), you should have this:

NOTE: if you specify bool, the SmartScope app will pick up all logic channels and analog channels (the app will convert them to binary values for you). If you specify float, the SmartScope app will pick up analog channels only. In case you want to build a decoder which consumes the output of another decoder, you can also specify DecoderOutput as input type.

Now you'll notice the inputWaveforms is a Dictionary, providing a typeless Array for each input you require. Before you can use the actual values inside the Arrays, you'll want to cast them to Arrays containing the type your defined earlier. In case of our previous example, this is how this would be done:

4.1 Decoder output

Now you can go ahead and implement your decoder. The output of your decoder must be an array of DecoderOutput elements.

You can have 2 types of DecoderOutput objects:

- the **DecoderOutputValue** serves to contain a decoded value (such as an address byte)
- the **DecoderOutputEvent** is used to store a decoded event (such as a Start condition).

When creating a DecoderOutput object, the first and second arguments you have to specify indicate the indices at which the visualization should start and end. The third argument allows you to specify the color in which it should be rendered to the screen.

Once you have defined the list of outputs for your decoder, transform it into an array and send it back to the SmartScope app, which will gladly display your newly created waveform. Additionally, other decoders will now also be able to select your decoder as input source.

5 Compile the project

Select Build -> Build solution (F6), solve any bugs and iterate until all bugs have been squashed.

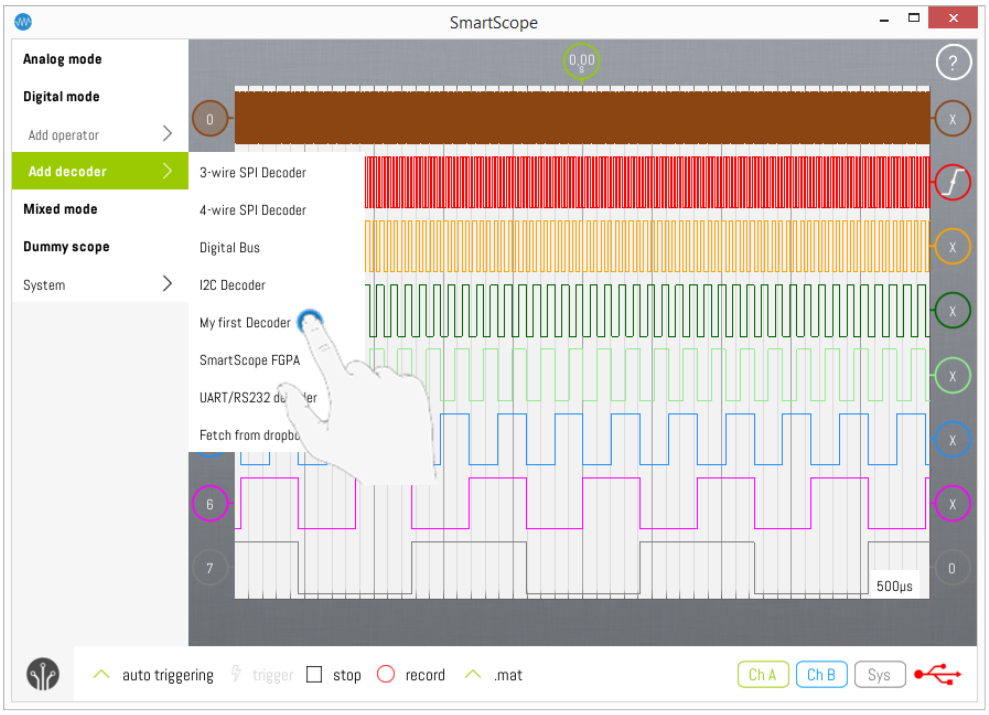
6 Move the .dll file to the right folder

Once compilation is finished, find the resulting .dll file. You can find the location where the .dll is being generated by right-clicking on your project (Decoders in the screenshot at the top of this page), selecting Properties, selecting Build in the menu on the left and then checking the Output path on the bottom of the page presented.

Copy or move that .dll file (Decoders.dll in case of this example) to the following folder:

Platform	Path
Mac	/Users/<username>/LabNation/Plugins
Linux	~/LabNation/Plugins
Windows	<My Documents>/LabNation/Plugins
Android	<sd-card>/LabNation/Plugins
iOS	See section below regarding DropBox

Now when you restart the SmartScope app, it should automagically pick up your new decoder and list it as available decoder!

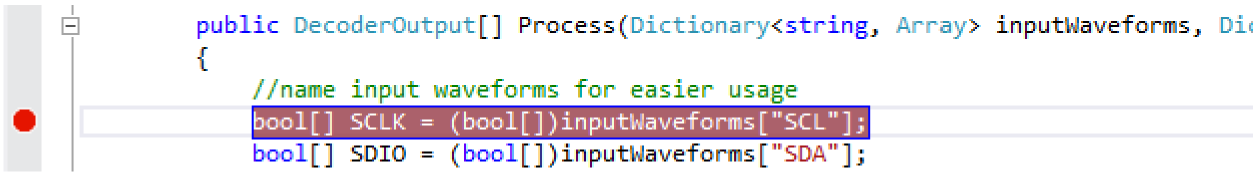


6.1 Using DropBox to transfer the dll to your tablet/phone

For all platforms, you can access a DLL file over DropBox. To do so, in the app go to Menu -> Add decoder -> Fetch from dropbox. If never done before, this will authenticate to DropBox and create all folders required. Next, on your PC you can save the DLL file to \Dropbox\Apps\LabNation SmartScope\Plugins. All DLL files you place here can now be accessed from all your devices over dropbox!

7 Debugging your decoder in real-time

Even though your project passes compilation successfully, there are many reasons why your decoder might not produce the desired result, or even cause the SmartScope throw an error message. The best way to figure out why, is to put a breakpoint in your code so you can step through your code while live data is being fed in. In order to do so, simply put a breakpoint at the first line of code of your Process method, as shown below. Start up the SmartScope app, and add your decoder.

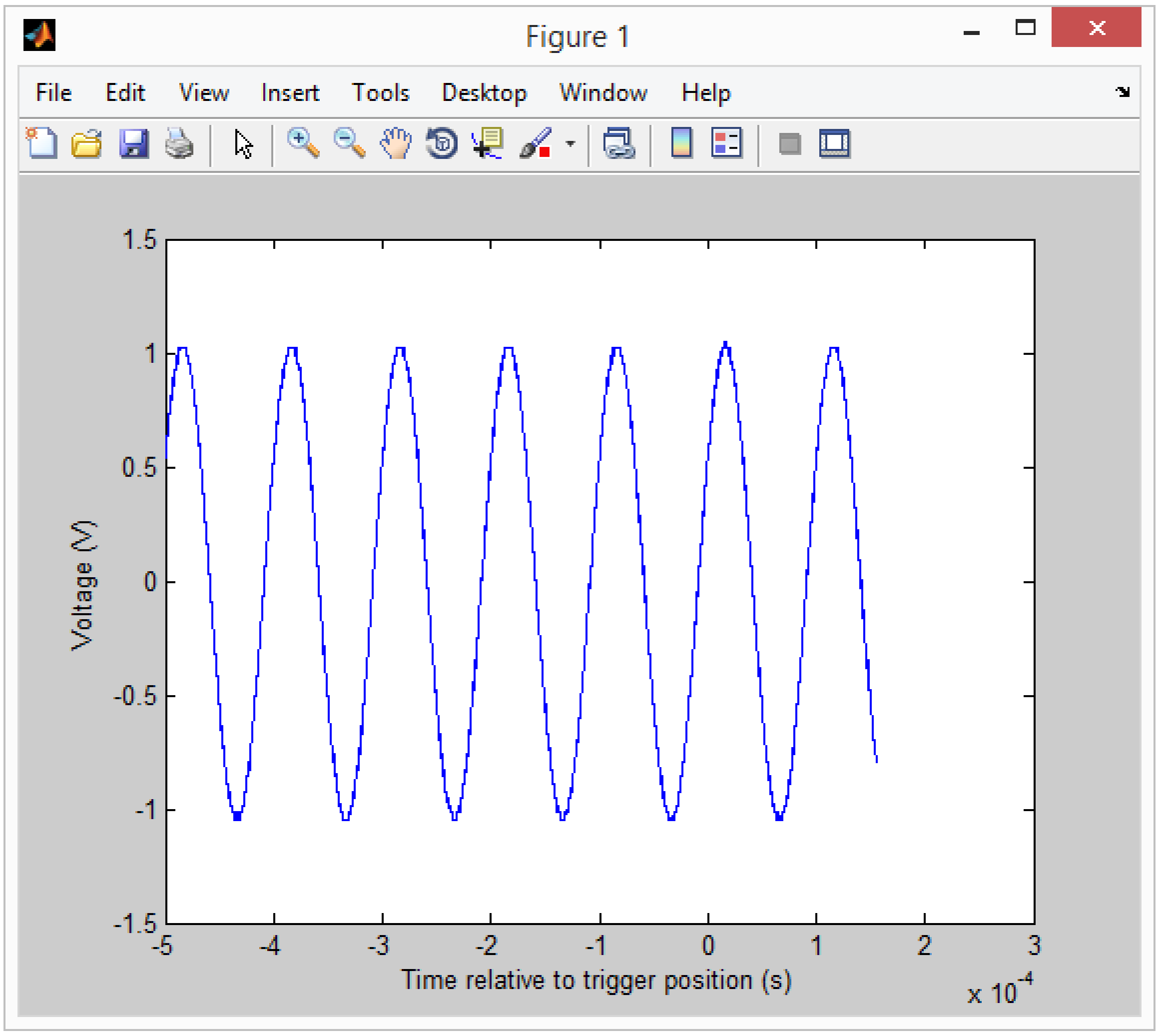


```
        public DecoderOutput[] Process(Dictionary<string, Array> inputWaveforms, Di
        {
            //name input waveforms for easier usage
            bool[] SCLK = (bool[])inputWaveforms["SCL"];
            bool[] SDIO = (bool[])inputWaveforms["SDA"];
```

Now usually you would expect the code to break at your breakpoint, but not in this case as the SmartScope app is using your compiled .dll, and not the code source you’re looking at. But no worries, we just need to let Visual Studio know that your source is exactly the same as the .dll which is currently being executed. To do so, select Debug -> Attach to process. Select SmartScope.exe from the list, and hit Attach. You should see that the SmartScope app is being halted, and your breakpoint is active now. Step through your code using F11 and F10 as you would debug any other program!

Controlling your SmartScope from Matlab

It is possible to configure all settings of the SmartScope from within Matlab, and to access all acquired data from within Matlab.
Data can be fetched and rendered to screen as shown below at **65fps on a regular laptop**.



Contents

1 Download the Matlab scripts

Start by downloading the matlab repository from our Github pages. Either clone the repository, or click the 'Download ZIP' button to download all files without any hassle.

2 Running the scripts

This should give you two files:

- SmartScopeConnect.m: start by running this script, which will connect to, and configure the SmartScope.
- SmartScopePlot.m: contains basic code which fetches the latest data from the SmartScope and draws it on the screen

The files themselves also contain some explanation, which is listed below.

2.1 SmartScopeConnect.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SmartScopeConnect - Connects and initializes the SmartScope using Matlab
%
% Prerequisites:
% - make sure the latest SmartScope software has been installed on this machine
% - make sure the first line of code refers to the correct location of LabNation.DeviceInterface.dll
% - make sure no other program is currently accessing the SmartScope
% - make sure a SmartScope is attached to this machine
%
% Usage:
% - just run the thing and you'll hear the SmartScope click hello
% - now hook up a 10kHz signal to ChA
% - run SmartScopePlot to see the results visualized in a graph
%
% Good to know:
% - you can only link to the .NET assembly and scope once (Matlab limitation). Therefore, that code is
enclosed inside the if clause below
%
% Tested on Matlab R2014a_64b+Windows8_64b
```

2.2 SmartScopePlot.m

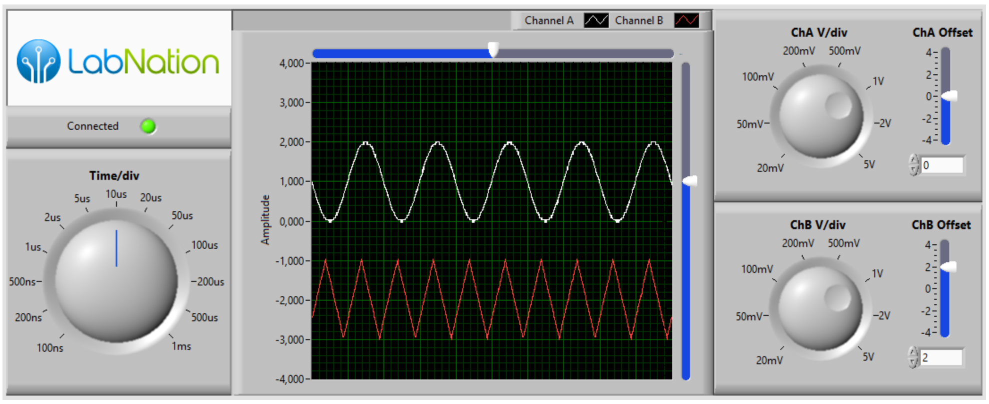
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SmartScopePlot - Fetches a couple of data sequences from the SmartScope
% and plots them in a graph. Also grabs some timing parameters.
%
% Prerequisites:
% - make sure SmartScopeConnect was executed succesfully first
%
% Usage:
% - just run this script
% - if 5000 frames are too much, hit Ctrl+C
%
% Good to know:
% - after running this script, type 'scope'
%   to see which parameters are exposed by the scope
% - after running this script, type 'scope.DataSourceScope.LatestDataPackage'
%   to see which parameters are exposed by the most recently acquired
%   datapackage
```

This page was last modified on 6 January 2016, at 21:05.

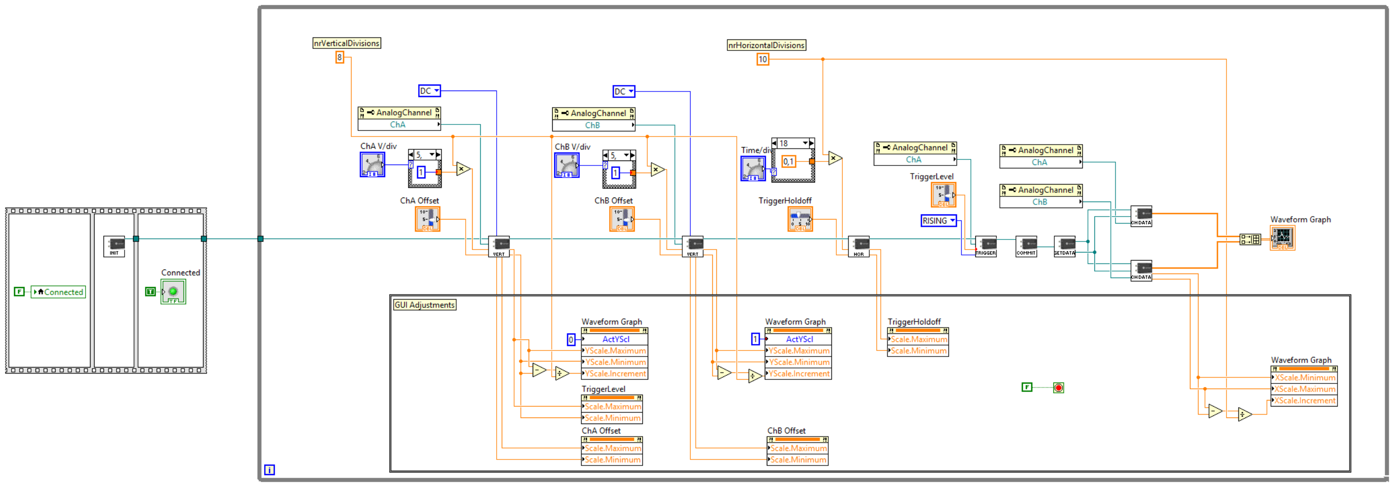
Controlling your SmartScope from LabView

LabView allows engineers to make responsive GUIs in hours, instead of weeks. Signal acquisition and generation devices can be bought from National Instruments, but they come at a cost. The SmartScope with its dual 100MS/s digitizers and 4MS of on-board RAM is an interesting alternative, and now comes with a set of VI blocks which allow you to configure and read out the SmartScope with ease.

This page explains all steps necessary to create the GUI shown below. Moreover, LabNation’s interface software is 100% open-source, and this page explains how to access more advanced features of this underlying library.



Behind this GUI is the block diagram shown below. Notice that the bottom part is only needed to update the GUI; all control and acquisition steps are done by the topmost part. All steps and VI blocks are be described below:



Contents

1 Downloading the SmartScope LabView VI's

Simply head to our LabView repo on GitHub and get those files on your PC. You can either clone the repo, or if you like it simple you can simply hit the 'Clone or Download' button and select Download .zip file. Extract these files to any location you want. The DemoGUI.vi is probably the file you'll want to try out first.

2 Using the SmartScope in a LabView application

When using the SmartScope in LabView, there are 3 phases to go through. This section lists them, together with all VIs for each phase.

2.1 Phase1: Initializing the SmartScope

This phase searches for a physical SmartScope, and returns a reference to the SmartScope which is needed by all other LabNation Vis.

1.1.1 INIT: Initialize.vi

When executed, this VI polls every 500ms whether a SmartScope is detected. This can be a SmartScope connected locally on the USB port, or a SmartScope shared over the network using the SmartScopeServer. Once detected, this VI will upload a basic configuration to the SmartScope and cause it to enter Running mode. The output of this VI is a reference to the SmartScope, and is needed for all other LabNation VI blocks. In practice, this means the dataflow in your block diagram will be blocked until a SmartScope becomes available.

2.2 Phase2: Configuring the Smartscope

At some point in your application, you will want to configure the acquisition settings of the SmartScope. This includes voltage ranges of the analog inputs, or the depth of the on-board RAM. It is important to note that all changes will only take effect after you've executed the CommitSettings.vi.

2.2.1 VERT: SetVertical.vi

This VI allows you to configure the analog input channels of the SmartScope. Below are the expected inputs:

- Voltage range: Simply pass it the largest amplitude you expect to measure, and the SmartScope will automatically change its dividing and multiplying stages in order to optimize its range for this amplitude.

Note that in order to get the finest resolution for your signal, it is best to set the voltage range as small as possible, while still making sure your entire signal can fit within that range.

- Offset: The value you specify here will be physically added or subtracted from the input voltage. This allows you to zoom in on a certain voltage range not centered around 0V. For example, if you want to measure signals between 0V and 12V, setting the voltage range to 12V and the offset to 6V will result in the highest measurement resolution.
- Coupling: This input terminal allows you to specify whether you want to have AC or DC coupling. DC coupling results in absolute voltages, while AC coupling first subtracts the mean value from the signal before being digitized. AC coupling allows to zoom in on signals with small amplitude but large offsets.
- Channel: last but not least, you should specify whether you're configuring Channel A or Channel B.

2.2.2 HOR: SetHorizontal.vi

This VI is a simplified version which allows you to define very basic timing settings. If you want to have more control over the timing settings including on-board RAM, see the last section of this page. Here are the required inputs:

- ViewportLength: allows you to define the length of the sequence to acquire, in seconds, and adjusts the sampling rate so the Viewport buffer of 2048 samples correspond to this length.
- TriggerHoldoff: specifies the position of the trigger, in seconds, relative to the center of the Viewport.

1.2.3 TRIGGER: AnalogTrigger.vi

This VI allows you to configure a basic analog trigger, using the following inputs:

- Channel: Which analog channel you want to trigger on
- EdgeType: This VI only supports Rising, Falling or Any edge.
- TriggerLevel: The voltage, when being crossed, causing a trigger event to occur

2.2.4 COMMIT: CommitSettings.vi

An understanding of this VI is quite important. Any changes you make through the previous VIs will have no effect until the CommitSettings VI is called. Using the previously described VIs, all changes you make are made in shadow registers, which are copied in 1 shot to the real registers only when you call the CommitSettings VI. This ensures acquisitions are made with a complete configuration.

2.3 Phase3: Reading out SmartScope data

At any moment, you can fetch the latest datapackage from SmartScope. A datapackage is a coherent set of information, containing the acquired data for all channels as well as information regarding the configuration used during the acquisition. It is important to have all of this data in one package, to ensure all data from different channels was acquired at the same moment in time.

2.3.1 GETDATA: GetLatestDataPackage.vi

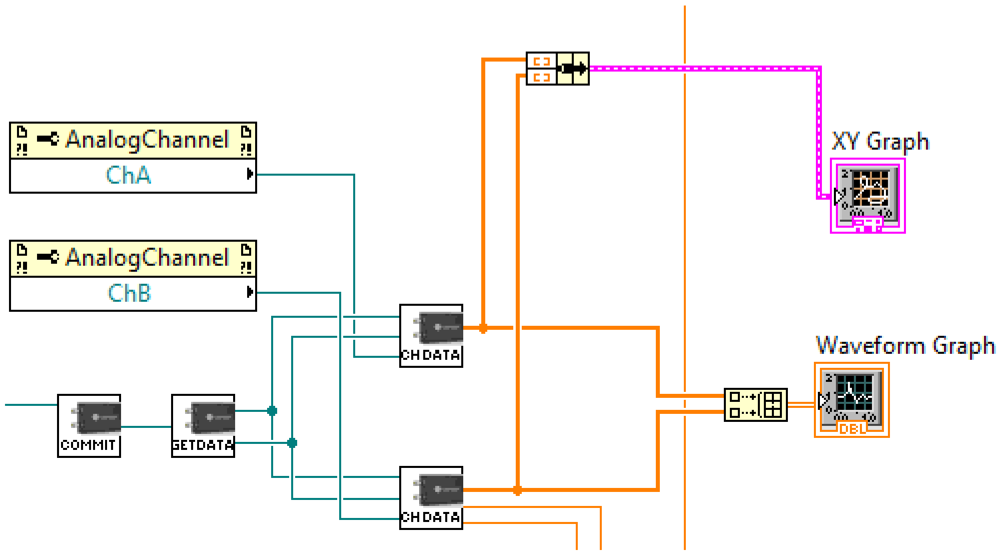
This VI simply fetches a reference to the latest datapackage received by the SmartScope driver. In case of acquisitions with a very long ViewportLength, it can happen that this VI is called multiple times during the same acquisition. In such case, the same reference to the last datapackage will be returned.

2.3.2 CHDATA: GetChannelData.vi

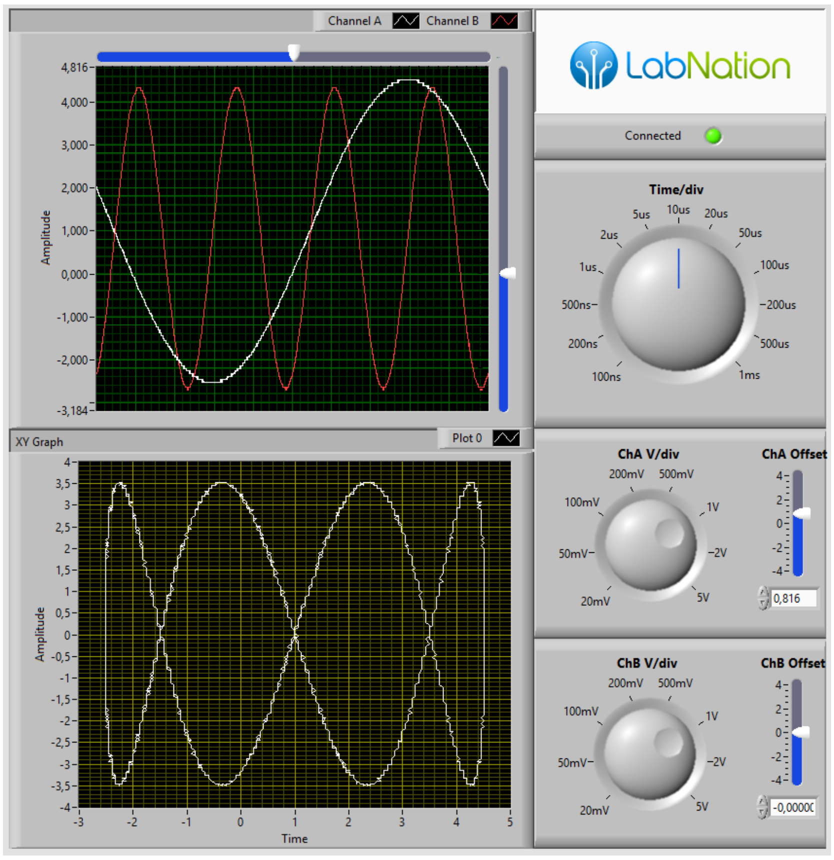
As explained above, a datapackage contains all data which was acquired at the same time for all channels. The GetChannelData VI simply returns the viewport data for the channel you specify, presented as a 1D array of floats. The Viewport data fetched by this VI are up to 2048 samples which span the duration you specified in the SetHorizontal VI. This VI requires a reference from the GetLatestDataPackage VI, as well as the analog channel you want to get the data from.

3 Adding a XY plot

With the SmartScope data available to your LabView program, it is incredibly easy to add new representations fast. For example, adding a XY plot is simply a matter of wiring in LabView's XY Chart VI:

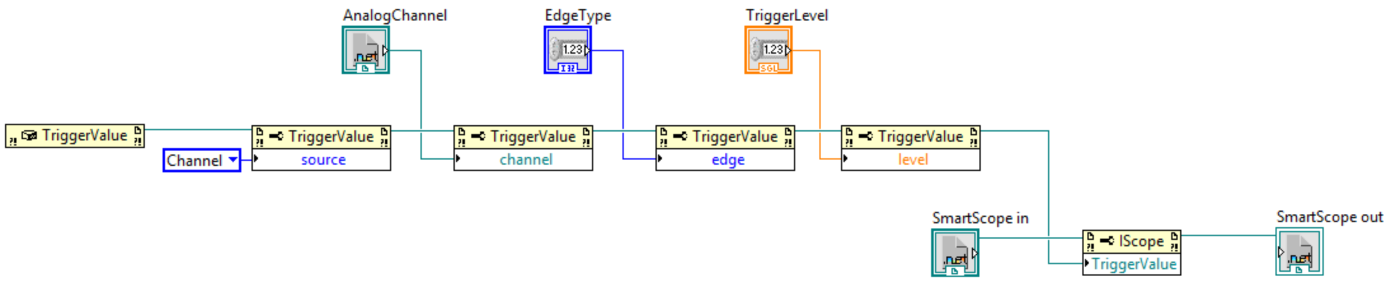


Which allows you to re-arrange the GUI easily:

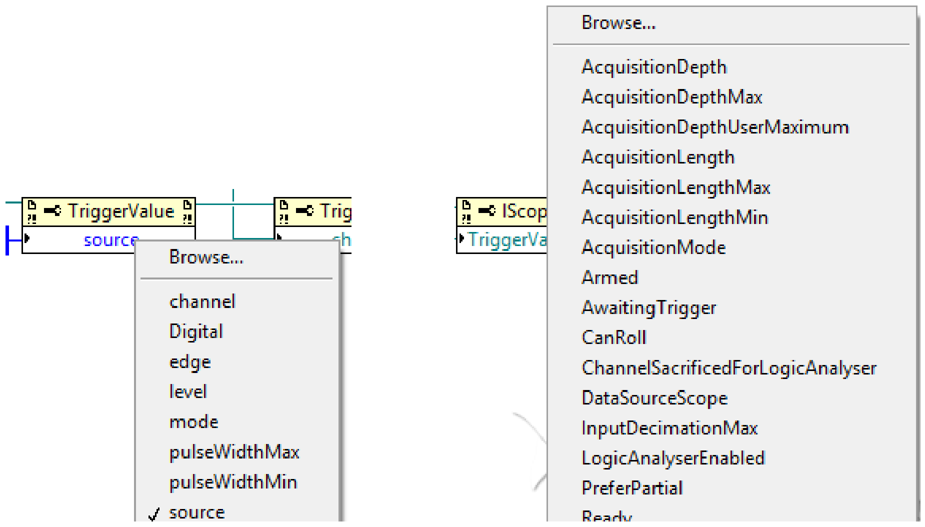


4 Accessing more advanced LabNation. DeviceInterface functionality

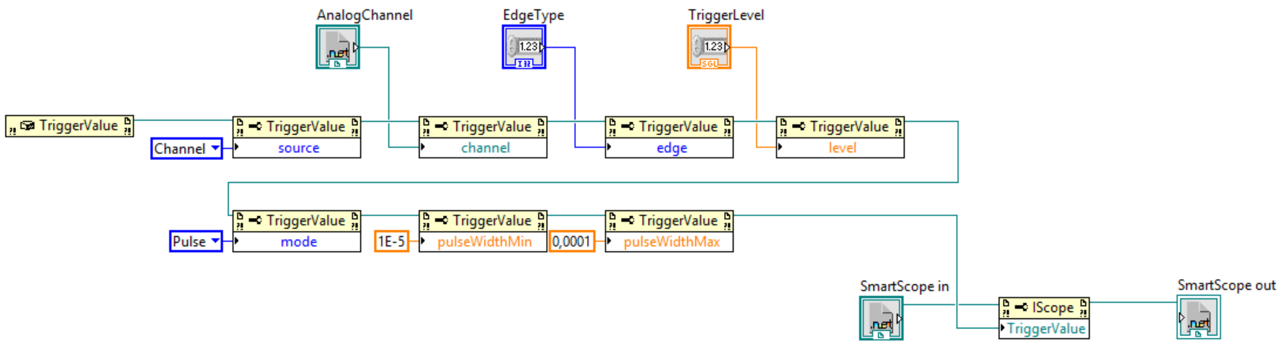
The software which interfaces with the SmartScope is 100% open-source. This DeviceInterface library is the only piece of code between your LabView environment and your SmartScope. As such, all functionality used by the official SmartScope app is accessible, also to LabView. Even though the VIs presented allows only basic operations, it is easy to use these more powerful features. <p>As an example, let's set up a slightly more advanced pulse-width trigger. The trigger should only fire when a pulsewidth between 10us and 100us is detected. While this would not be possible using the VIs described above, the SmartScope supports this through the DeviceInterface, and hence we can use it from LabView. And it's not all that difficult. <p>To start, open up the AnalogTrigger VI and copy-save it to AnalogTriggerPulse. Have a look at the block diagram:



You see this diagram contains new blocks. LabView takes the DeviceInterface library, and automatically creates these blocks for all functions inside the library. To give you an idea, simply click on any Property of a TriggerValue block, eg on 'source'. You will see a nice drop-down list of all properties you can set. Click also on the 'TriggerValue' Property of the IScope object on the bottom-right: you'll get a list of all Properties of the SmartScope you can set. These are explained on the DeviceInterface section of the LabNation wiki, and 100% of the source can be found on LabNation's GitHub account.



If we want to change the trigger from simple edge-detection to pulse detection, simply add the 3 blocks shown below. You can simply copy-paste the blocks above, and change their properties to ‘mode’, ‘pulseWidthMin’ and ‘pulseWidthMax’. <p>To create the ‘Pulse’ constant, simply right-click on the ‘mode’ terminal, select Create->Constant and LabView will automatically present you a list of the various possibilities. The other 2 constants define the minimum and maximum pulse width, which are now fixed but you can easily make them adjustable by using a Control instead of a Constant.



Voila, when you use this new subVI in your main VI, your SmartScope will reject all triggers which are shorter than 10us or longer than 100us.

Using the smartscope on the network

The SmartScope can be used over a network connection, i.e. a WIFI. This allows you to use the smartscope wirelessly. We currently provide a software package, SmartScopeServer, which can run on any Win/Lin/Macos device. Upon connecting a SmartScope, the SmartScopeServer broadcasts this over the network. Next, take your tablet, start our normal app, and it will automatically detect the SmartScopeServer and take control over the SmartScope. This allows you to control the smartscope wirelessly, at almost identically the same performance as if it was connected with a wire to your tablet.

Current benefits of the SmartScope server:

1. Allows to control a SmartScope from a (non-jailbroken) iDevice. Simply download the SmartScope app from the regular AppStore, and it will connect to a SmartScopeServer running in the network!
2. Useful in case you need to use your laptop for other documents (schematics): connect the SmartScope to the SmartScopeServer running on your laptop, and you can control the SmartScope from any tablet.
3. In case your tablet is running low on battery, you can use its USB port for charging
4. Leave your SmartScope at a test setup, and you can visualize the result from your chair
5. ...

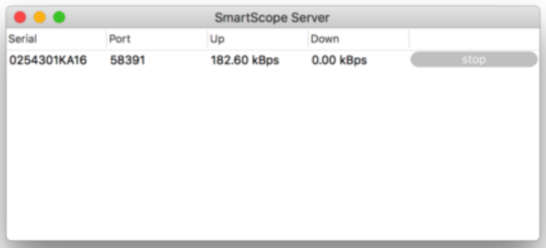
As a next step, we are porting this server to the Smartscope WiFi Bridge, a lightweight wifi router with battery; which will then provide a truly wireless battery-powered solution. The server software will be running on this device, acting as host to other tablets/phone/pcs on the network running the SmartScope user interface.

***Note:** the SmartScope WiFi Bridge is currently only available to a selected group of beta testers.*

To use the smartscope over the network today (with the software-only solution), it needs to be connected to a host which can run the SmartScope Server (which can be any pc, including RaspberryPi). This host then needs to be on the same LAN as the device on which you want to control the SmartScope through the app.

On Win/Lin/Macos, the **SmartScopeServer is contained in the same installer package as the regular SmartScope app, and installed into the same folder**. A console and UI version exist of the server.

Mac

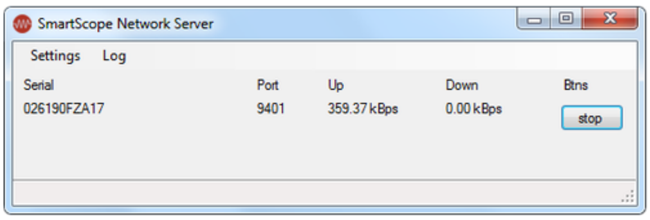


SmartScope Server

Serial	Port	Up	Down	
0254301KA16	58391	182.60 kBps	0.00 kBps	stop

1. Run the SmartScope server UI from SmartScope DMG package
2. Connect a SmartScope to your Mac
3. A new line should appear in the SmartScope Table. Press the start button, to start the server
4. Run the app on your other device, which is on the same LAN as the windows host running server
5. Connection should happen automatically

Windows



SmartScope Network Server

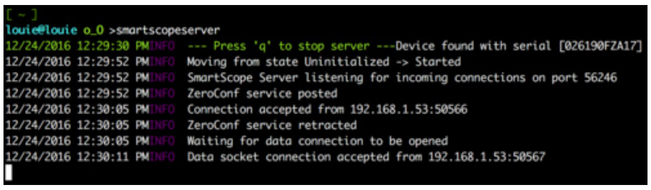
Settings Log

Serial	Port	Up	Down	Btns
026190FZA17	9401	359.37 kBps	0.00 kBps	stop

1. Install the latest SmartScope software package from our download page

2. Run the SmartScope server UI from the start menu, or from C:\Program Files (x86)\LabNation\SmartScope\SmartScopeServerUI.exe
    1. In case an error regarding ZeroConf pops up, this means you don't have any ZeroConf service installed. Google for 'Bonjour Print Services for Windows' to make sure you download the latest version from Apple, and install this package.
3. Connect a SmartScope to this windows machine
4. Press the start button, to start the server
5. Run the app on your other device, which is on the same LAN as the windows machine running server
6. Connection should happen automatically

Linux



```
[ ~ ]
louie@louie o_0 >smartscopeserver
12/24/2016 12:29:30 PMINFO  --- Press 'q' to stop server ---Device found with serial [026190FZA17]
12/24/2016 12:29:52 PMINFO  Moving from state Uninitialized -> Started
12/24/2016 12:29:52 PMINFO  SmartScope Server listening for incoming connections on port 56246
12/24/2016 12:29:52 PMINFO  ZeroConf service posted
12/24/2016 12:30:05 PMINFO  Connection accepted from 192.168.1.53:50566
12/24/2016 12:30:05 PMINFO  ZeroConf service retracted
12/24/2016 12:30:05 PMINFO  Waiting for data connection to be opened
12/24/2016 12:30:11 PMINFO  Data socket connection accepted from 192.168.1.53:50567
```

By installing the .deb package, the SmartScope Server CLI binary is placed in **/opt/smartscope/SmartScopeServer.exe**. A script-link to this binary is also placed in **/usr/bin/smartscopeserver**. Also a desktop entry for the Server UI is made in your desktop environment's Application menu, in the Science category.

To run the console version, on a terminal prompt, type

# smartscopeserver

If this doesn't work for some reason, you can also try to explicitly start mono with the binary as an argument

# mono /opt/smartscope/SmartScopeServer.exe

Similarly, you can run the UI version from the command line using

# smartscopeserverui

or

# mono /opt/smartscope/SmartScopeServerUI.exe

This page was last modified on 21 June 2017, at 14:10.

In case of a crash: please send in the CrashReport!

We're not proud of crashes. On the contrary, we do everything possible to fix and avoid them. Please help us and fellow users by sending in the crash report, automatically generated in the unfortunate event of a crash.

Crash Reports

Since v0.9.0.0, the SmartScope app detects when a crash is about to occur, and saves all details it has about the upcoming crash into a file on disk.

Please send in your CrashReports to bughunt@lab-nation.com

Before making a public release of a new software version, our app goes through a cycle of validation tests. We don't make public releases which contains known bugs. However, since we're targeting a very wide range of platforms, it is clear we cannot test on every device available. Therefore, IF you experience a crash, the information in the resulting CrashReport is very valuable to us. Please be so kind to send it to **bughunt@lab-nation.com**, so it enters our bugtracker which allows us to provide a fix as soon as possible in the beta channel.

(we do not want to transmit these crashreports automatically from your PC to our servers)

Location of the CrashReports

Here is the location where you can find the generated CrashReports:

Platform	CrashReport path
Mac	/Users/<username>/LabNation
Linux	~/LabNation
Windows	<My Documents>/LabNation
Android	<sd-card>/LabNation

Contents of each CrashReport

Each CrashReport gives an accurate pointer to the location in our code where the crash originated from (the stacktrace). Additionally, it gives us some information which helps us to reproduce the bug, like the version of the SmartScope software, and the operating system you're currently using.
Below you can see such a sample CrashReport:

```
-----------------------------------------------------------------------------------------
------------------------------          CRASH REPORT         ------------------------------
Timestamp: 16/06/2016 21:44:26
SmartScopeApp version: 0.0.6011.37327
OS: Windows Microsoft Windows NT 6.2.9200.0
Error message: Destination array is not long enough to copy all the items in the collection. Check array
index and length.
Source: System.Core
TargetSite: CopyTo
StackTrace:    at System.Collections.Generic.HashSet`1.CopyTo(T[] array, Int32 arrayIndex, Int32 count)
   at System.Collections.Generic.List`1..ctor(IEnumerable`1 collection)
   at System.Linq.Enumerable.ToList[TSource](IEnumerable`1 source)
   at LabNation.DeviceInterface.DataSources.ChannelDataSource.get_List() in
c:\LabNation\SmartScopeApp\DeviceInterface\DataSources\DataPackageScope.cs:line 17
   at ESuite.ScopeDataCollection..ctor(DataPackageScope scopeData) in
c:\LabNation\SmartScopeApp\ESuite\ScopeDataCollection.cs:line 38
   at ESuite.EMainEngine.UpdateNewestDataPointer(DataPackageScope scopeData, DataSource dataSource) in
c:\LabNation\SmartScopeApp\ESuite\EMainEngine.cs:line 70
   at LabNation.DeviceInterface.DataSources.NewDataAvailableHandler.Invoke(DataPackageScope dataPackage,
DataSource dataSource)
   at LabNation.DeviceInterface.DataSources.DataSource.fireDataAvailableEvents() in
c:\LabNation\SmartScopeApp\DeviceInterface\DataSources\DataSourceScope.cs:line 31
   at LabNation.DeviceInterface.DataSources.DataSource.DataFetchThreadStart() in
c:\LabNation\SmartScopeApp\DeviceInterface\DataSources\DataSourceScope.cs:line 114
   at System.Threading.ExecutionContext.RunInternal(ExecutionContext executionContext, ContextCallback
callback, Object state, Boolean preserveSyncCtx)
   at System.Threading.ExecutionContext.Run(ExecutionContext executionContext, ContextCallback callback,
Object state, Boolean preserveSyncCtx)
   at System.Threading.ExecutionContext.Run(ExecutionContext executionContext, ContextCallback callback,
Object state)
   at System.Threading.ThreadHelper.ThreadStart()
-----------------------------------------------------------------------------------------
```

SmartScope - Debug checklist

In case something happened to your SmartScope, you can send in your SmartScope for repair, or (since you're into electronics anyway) you might try to give it a shot yourself. This document gives you a starting point in pinpointing what might be the problem.

Checking for short-circuits

The first thing to check is whether some power rails might be shorted. This might happen when a component was exposed to excessive currents. Check whether none of the nets in the table below are connected to each other, by checking either the bottom-left or top-right part of the table below. Make sure you the image below to find out where to probe the nets.

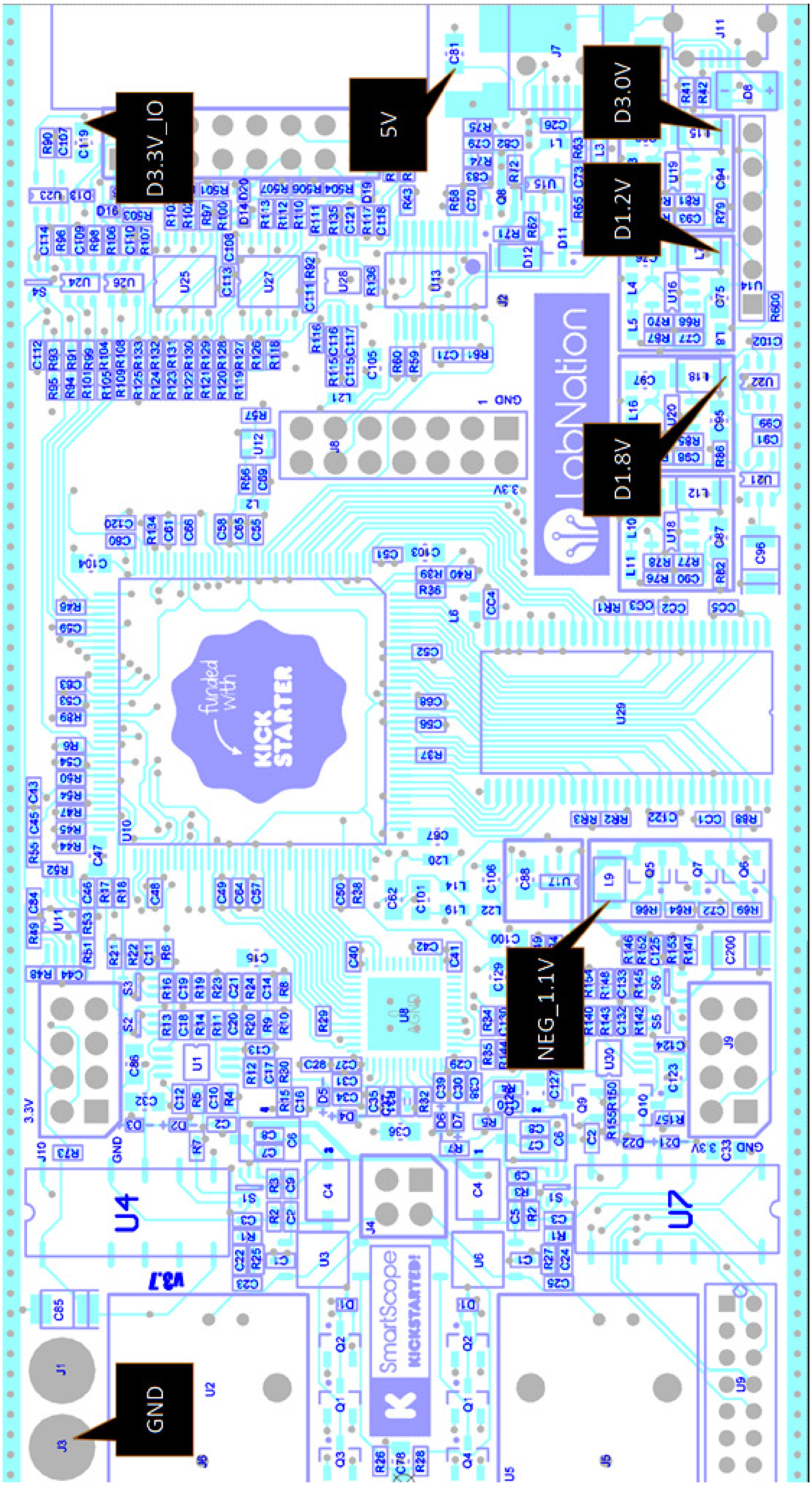
	GND	5V	D3.3V_IO	D3.0V	D1.8V	D1.2V	NEG_1.1V
GND							
5V							
D3.3V_IO							
D3.0V							
D1.8V							
D1.2V							
NEG_1.1V							

Verifying supply rail voltages

Next, power the SmartScope and measure all voltages to see whether they're close enough to their expected value.
Note: in the app, switch to Digital mode to power the D3.3V_IO net

	Measured voltage
GND	
5V	
D3.3V_IO	
D3.0V	
D1.8V	
D1.2V	
NEG_1.1V	

This page was last modified on 1 March 2017, at 11:49.


D3.3V_IO
5V
D3.0V
D1.2V
D1.8V
NEG_1.1V
GND

HARDWARE (Parcial)

ADC resolution - voltage dividing/multiplying stage

General

When you are using ranges (20mV/div - 100mV/div) it uses a different path than when using (200mV/div or higher), this is to reduce noise and to get more resolution. You'll always hear the relay switching in the transition between those ranges.

On each voltage range (vertical resolution in volts/div) you will have different minimum and maximum value that the scope can measure.

This depends on your current vertical offset and there is some extra room out of the screen (that is different on each range) but as general rule of thumb if a signal goes outside of the screen vertically you are probably exceeding the limits of that range and the scope is clipping the signal.

So you need to adjust the resolution and vertical offset so you can see the whole signal in the screen (vertically) if you want to measure it.

Example for a 2V signal

Since the main grid on the SmartScope software has 8 division and your signal is 2V that means that 200mV/div x 8 divisions = 1.6V is not enough, and that is why your voltage got clamped to 1.4V, probably your were using an offset of -3 division giving you only 7 div. x 200mV = 1.4V! so your signal was clamped to 1.4V

And in lower ranges it will be even worse, but that is the normal operation of any scope, you cannot measure larger signals than what you can fit in the screen in your current range.

Voltage resolution

The resolution of the scope also varies between ranges, so don't expect to see exactly the same value in each range, but they should be really close.

The resolution of the ranges is displayed in the next table:

Range Max.	Res. Max.	Res. Typ.
20 mV/div	1.6 mV	2 mV
50 mV/div	1.6 mV	2 mV
100 mV/div	3.2 mV	5 mV
200 mV/div	6.3 mV	10 mV
500 mV/div	17 mV	20 mV
1 V/div	32 mV	50 mV
2 V/div	63 mV	100 mV
5 V/div	157 mV	200 mV

In addition to that, different ranges use different amplification so noise, temperature and small calibration errors could make also the values differ even more.

And after all, this is an 8-bits scope, it is not intended for precise voltage measurement, so you can be confident that it will give you an accuracy of arround 10% of your current volts/div but do not ask for more wink. So in 1V/div the figures will be accurate to 0.1V, and in 500mV/div to 0.05V.

This page was last modified on 15 July 2017, at 14:22.