Siglent SDS1104X-E Review

Advanced Functions

Math

The math functions (other than FFT) are not exactly a strong point of the SDS1104X-E. They are pretty basic and I have to admit that I personally only use a few of them: Add, Subtract, Multiply, Integral and FFT. Other than that, I don't consider single operator math functions particularly useful, no matter how many of them were available.

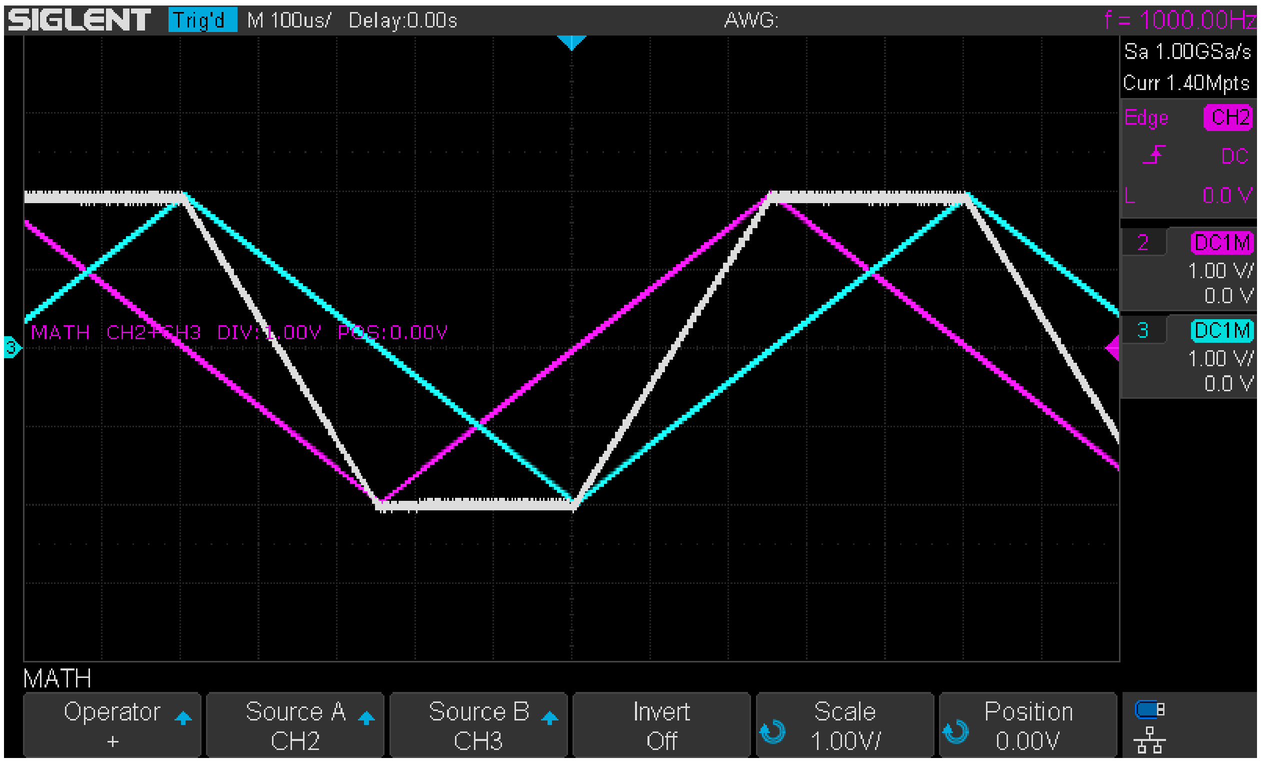
Whenever I have a need for math channels, then I might use more than one of them at the same time, want to be able to enter complex formulas and expect heavy support through a comprehensive library of math functions. None of these is available in the SDS1104X-E and this situation is not helped by the rather loveless implementation of most math operators, which often yields rather ugly, grainy and noisy results.

Add & Subtract

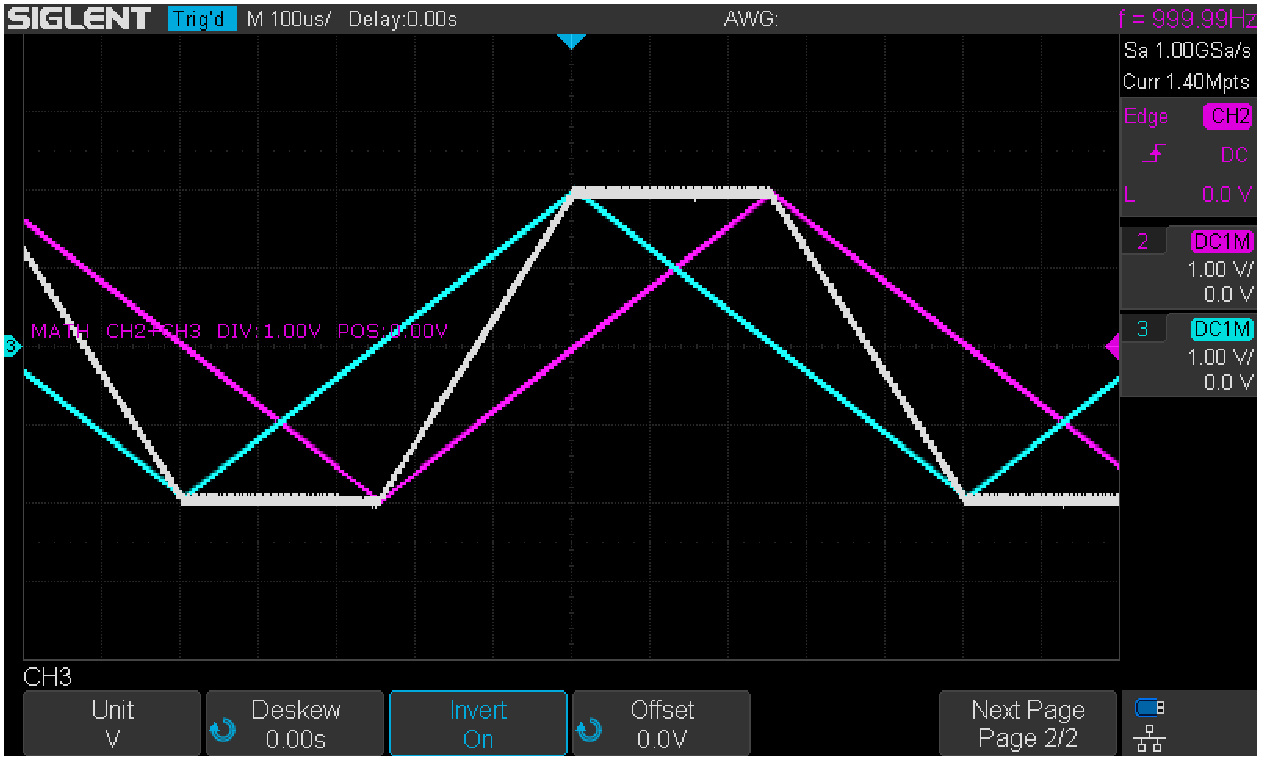
This is something even old analog scopes could do. In fact, it was only *Add* and a subtraction was accomplished by inverting the 2nd channel. There was also no dedicated math channel but the original traces were simply replaced by the sum of the two input channels.

Like most other DSOs, the Siglent SDS1104X-E has a dedicated math channel that is displayed together with the original input channels. The math channel has a white trace which doesn't look particularly pretty, as it is fat and grainy in most situations.

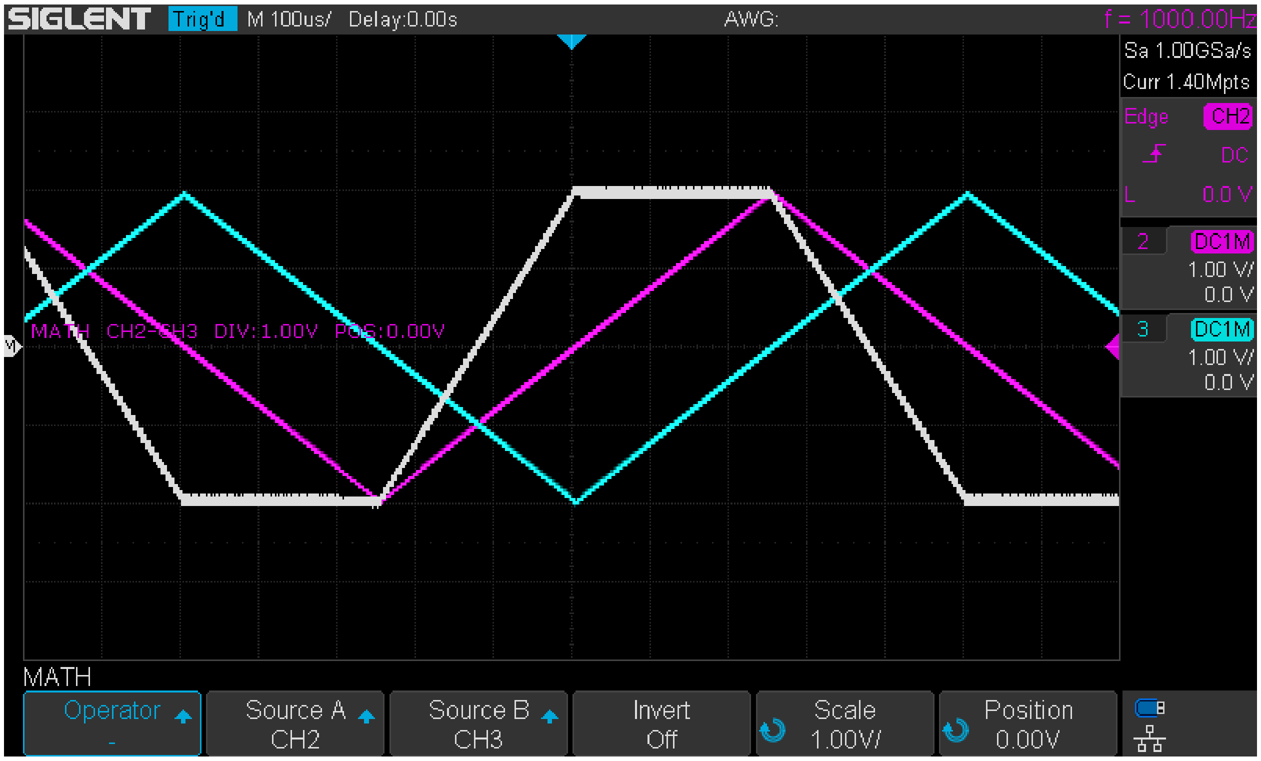
First is the sum of channels 2 + 3.
Second is the sum of channels 2 + inverted 3, to mimic the subtraction on an analog scope.
Third is the difference of channels 2 – 3, which should give the exact same result – and it certainly does.



SDS1104X-E_Add



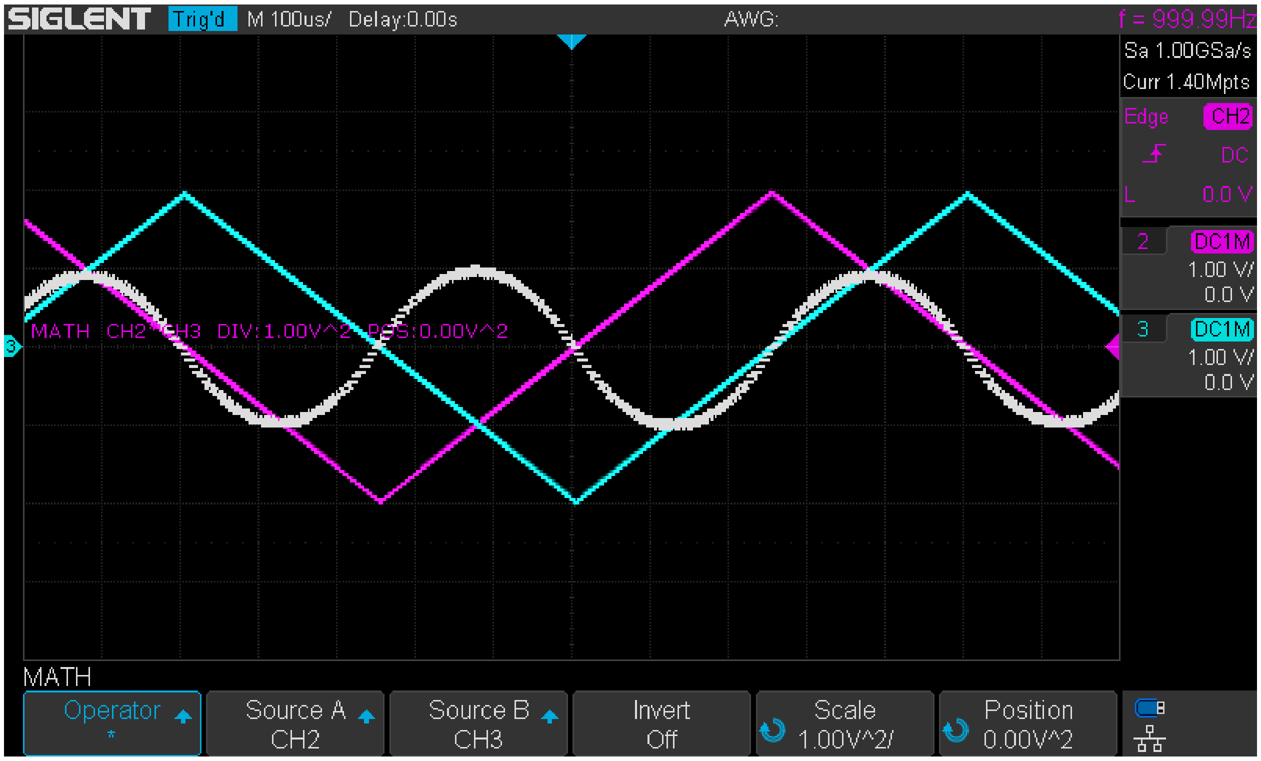
SDS1104X-E_Add_InvB



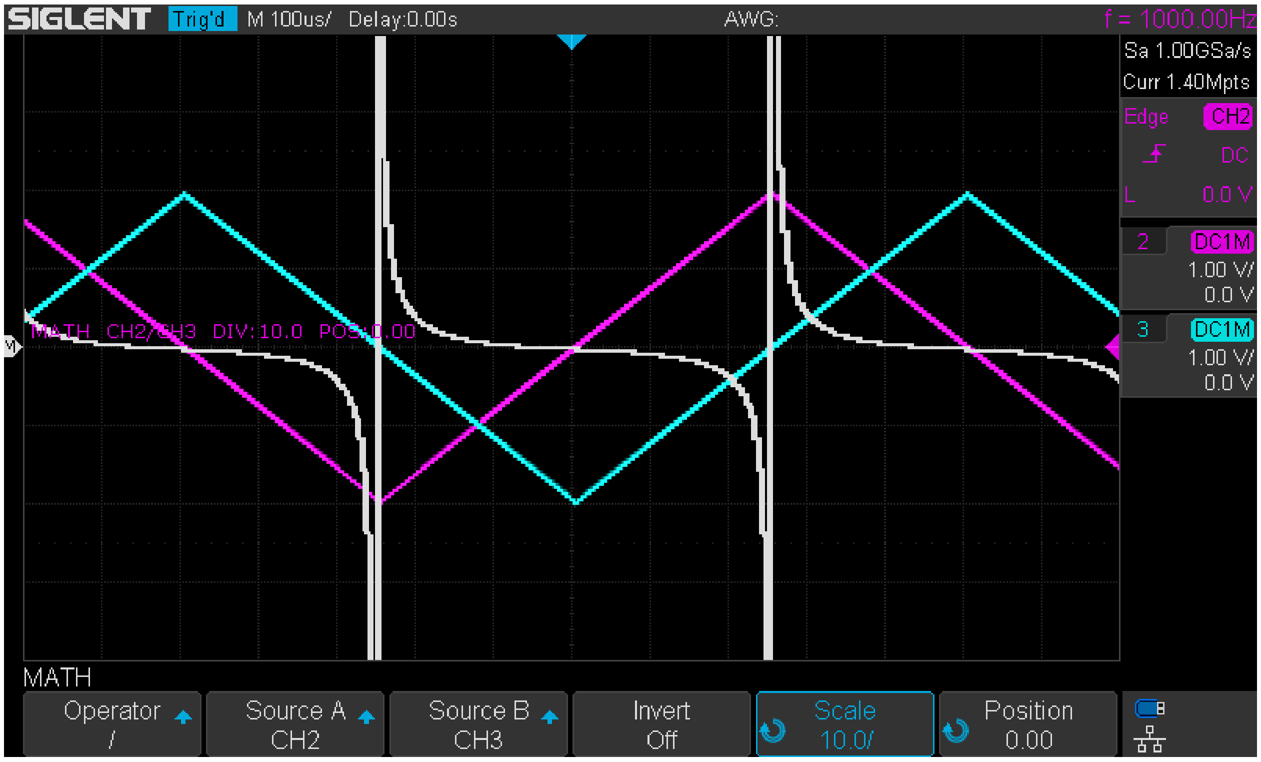
SDS1104X-E_Sub

Multiply & Divide

This could be used for signal gating as well as generating sum and difference frequencies – or producing a sinusoidal waveform out of two triangles, as in the example below.



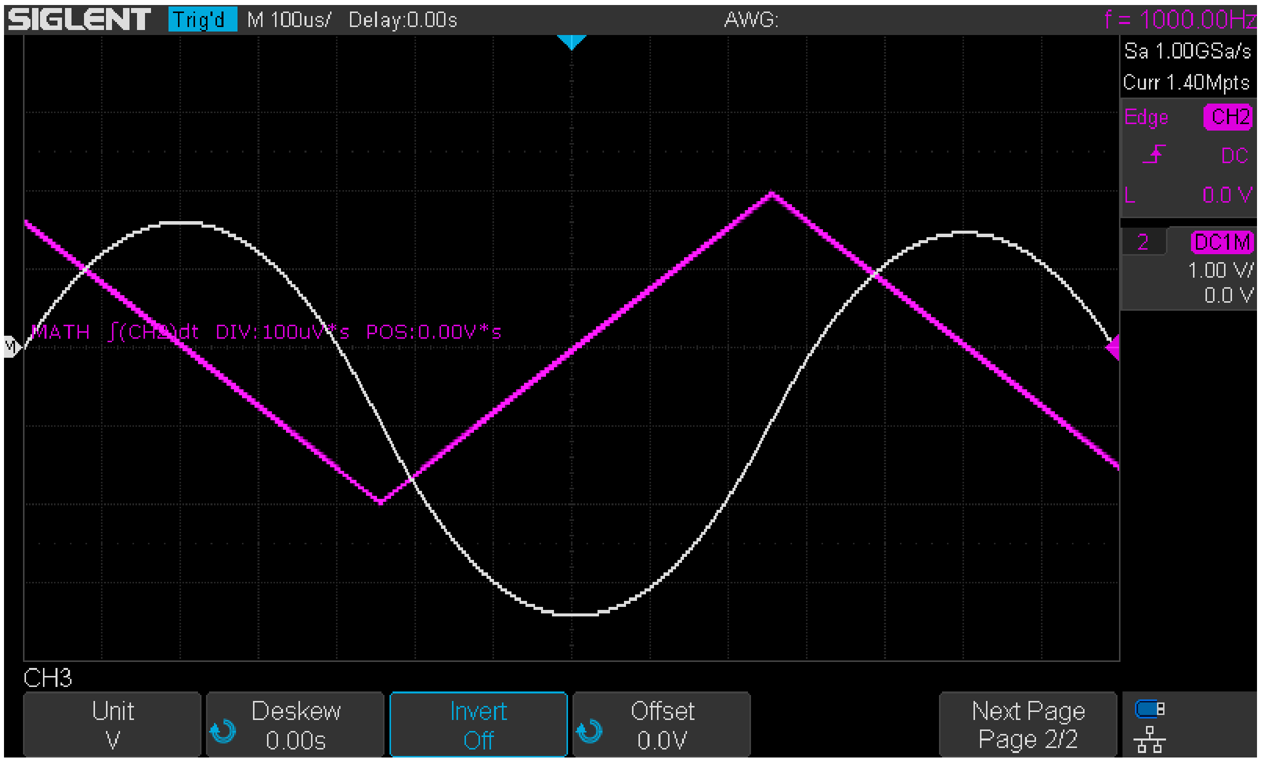
SDS1104X-E_Mul



SDS1104X-E_Div

The division as shown above looks reasonably nice and at least the scope is not thrown off track when the denominator becomes zero.

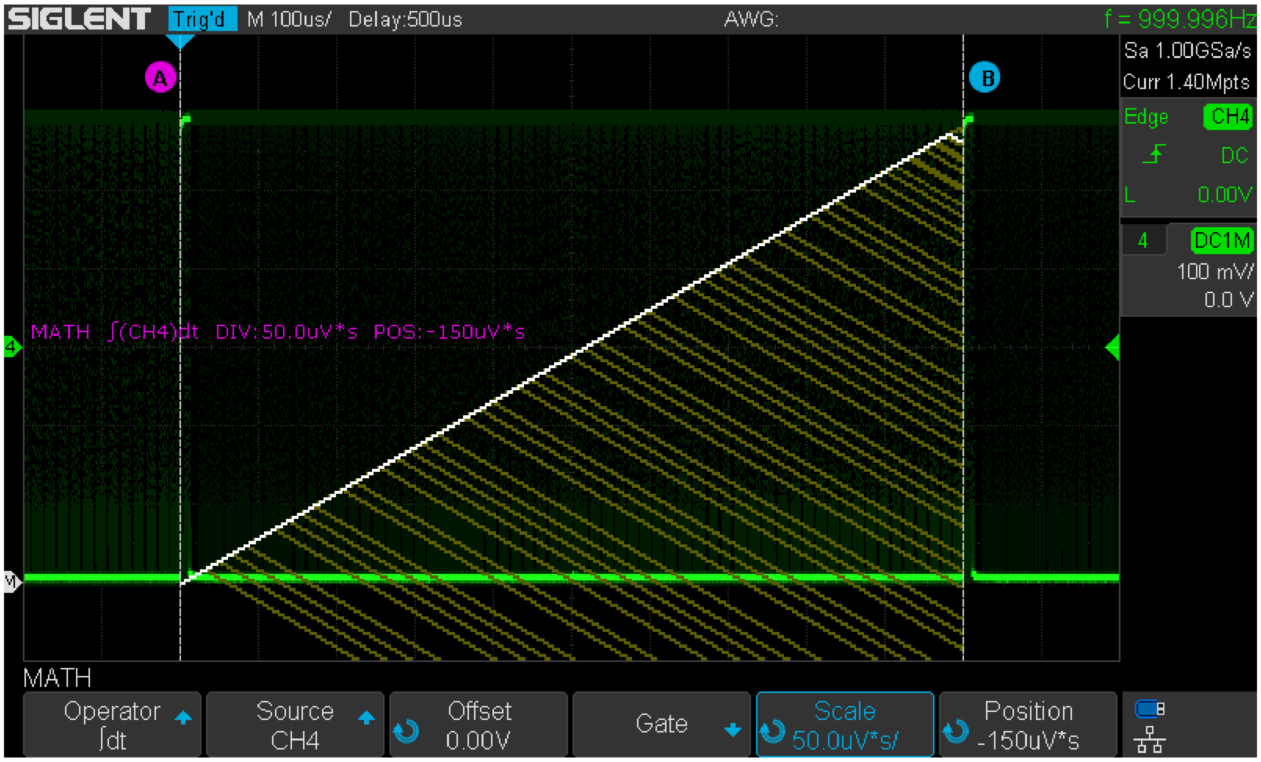
Integral



SDS1104X-E_Int

The integral function shown in the example above works very well by attenuating the harmonics of the triangle wave and thus turning it into a sinusoidal waveform.

The integral function is somewhat special in that it has an implicit gate function to calculate a definite integral. This can be used to display the output of a PWM signal.



Integrate_1ms_persistence_10s

In the example above, a PWM signal has been swept between 0 and 100% duty cycle and the gate for the integration has been defined for the PWM period. 10s persistence has been used in an attempt to visualize the dynamic signal in a static screenshot, which has been taken just before the 100% duty cycle had been reached. The white trace can be interpreted as the output voltage as a function of the duty cycle. The yellow lines are just previous math traces still visible because of the persistence.

Differential

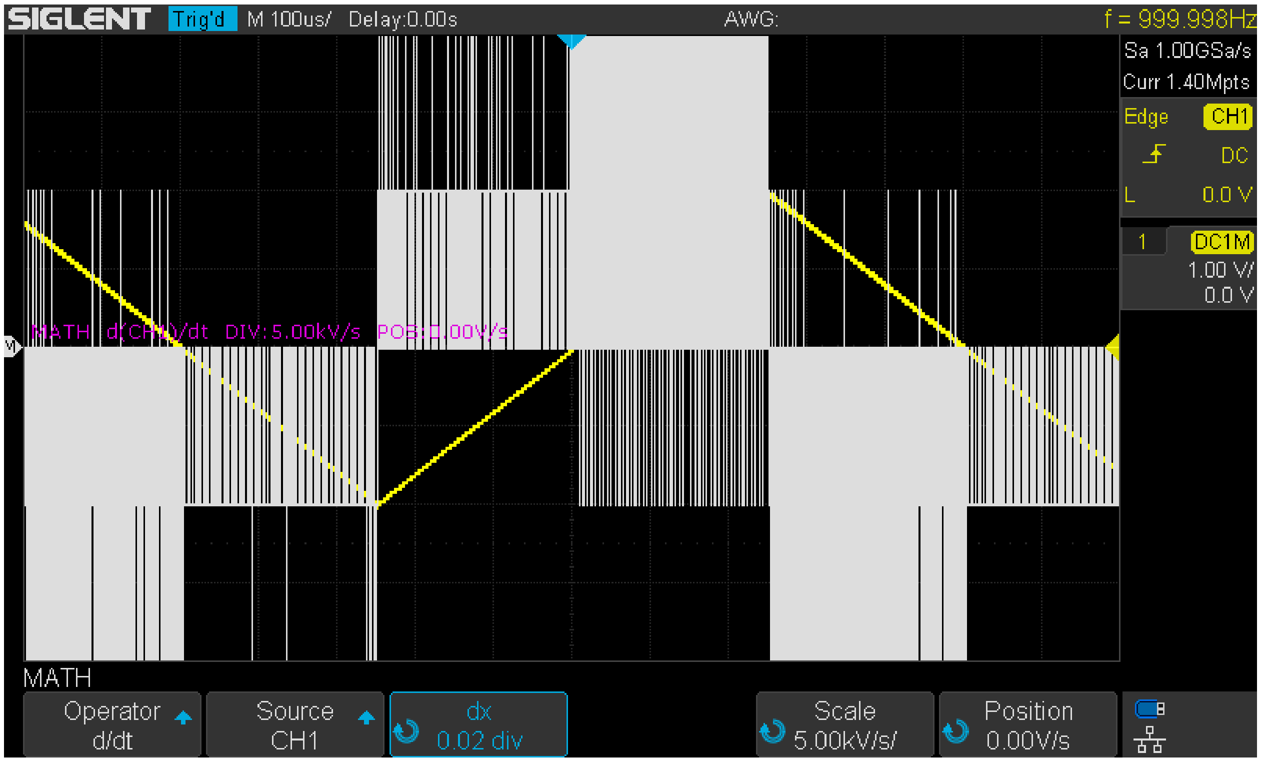
This is a particularly cumbersome math function that just cannot work well on an 8 bit system. It has a parameter *dx*, which determines the interval used for the difference computation. For an accurate and detailed result, we would want *dx* to be as small as possible and the lowest value that can be set is 0.02 div. With this setting, we can get totally useless results, as we will see later.

Why is it so? Let's do some simple math.

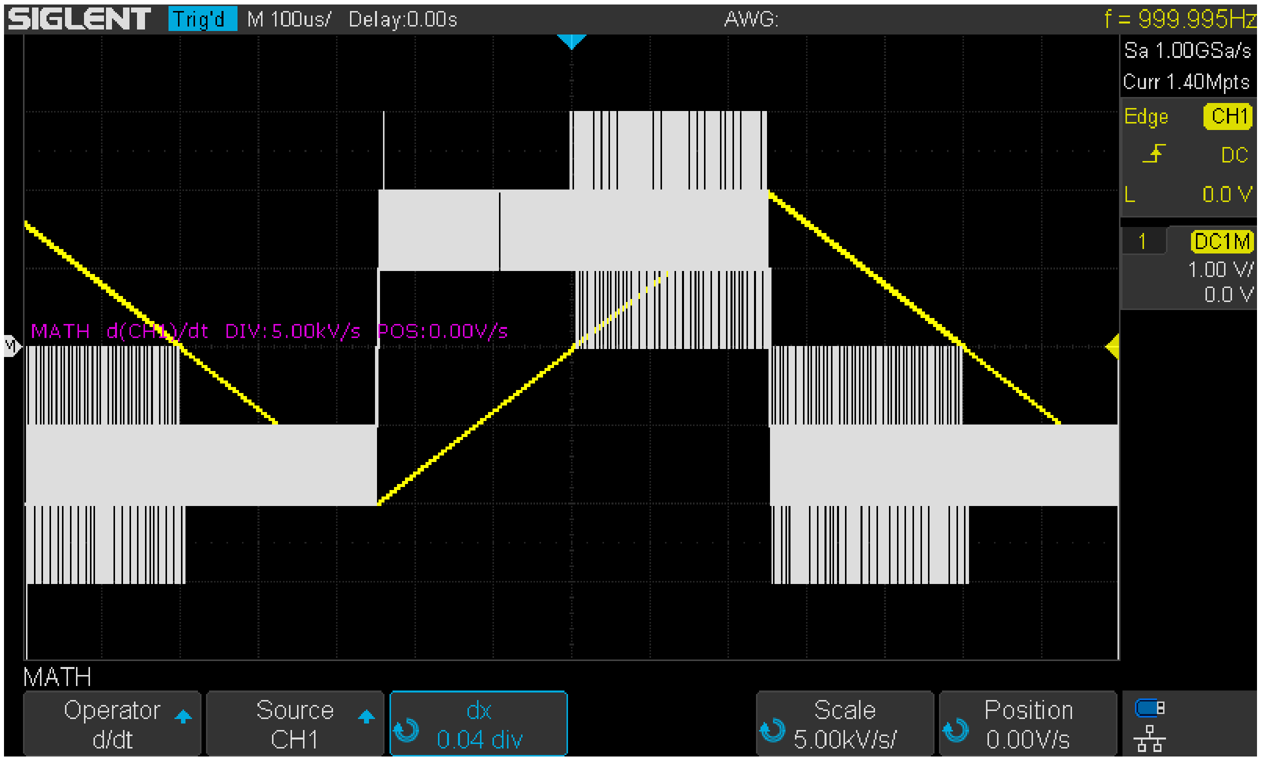
The screen is 800x480 pixels, with 700x400 dedicated to the trace area. Since there are 14 horizontal divisions, one division is 700/14 = 50 pixels wide. The lowest *dx* parameter would thus be 0.02 x 50 = 1, which means the difference between two adjacent samples is calculated. This will work well for fast transitions, but not for a triangle wave, which is about the worst we can throw at the differentiate function.

For the amplitude, the trace area on the screen shows 200LSB of the ADC, and there are 8 vertical divisions, hence we get 200/8 = 25 LSB per division.

Let's have a look at a triangle (symmetrical ramp) with 4Vpp and 1kHz. A triangle should turn into a square when differentiated. One ramp (up or down) takes 500µs or 5 divisions at 100µs/div timebase and the amplitude is 4 divisions at 1V/div vertical gain. For a slope of 1 as with the triangle, one horizontal pixel (equivalent to *dx*=0.02) corresponds to half a LSB in vertical direction. This quite obviously cannot work and even *dx*=0.04 means just one LSB per time interval, where we are down in the DNL (differential nonlinearity) and noise. So no wonder that low *dx* parameter settings don't work with slow ramps.

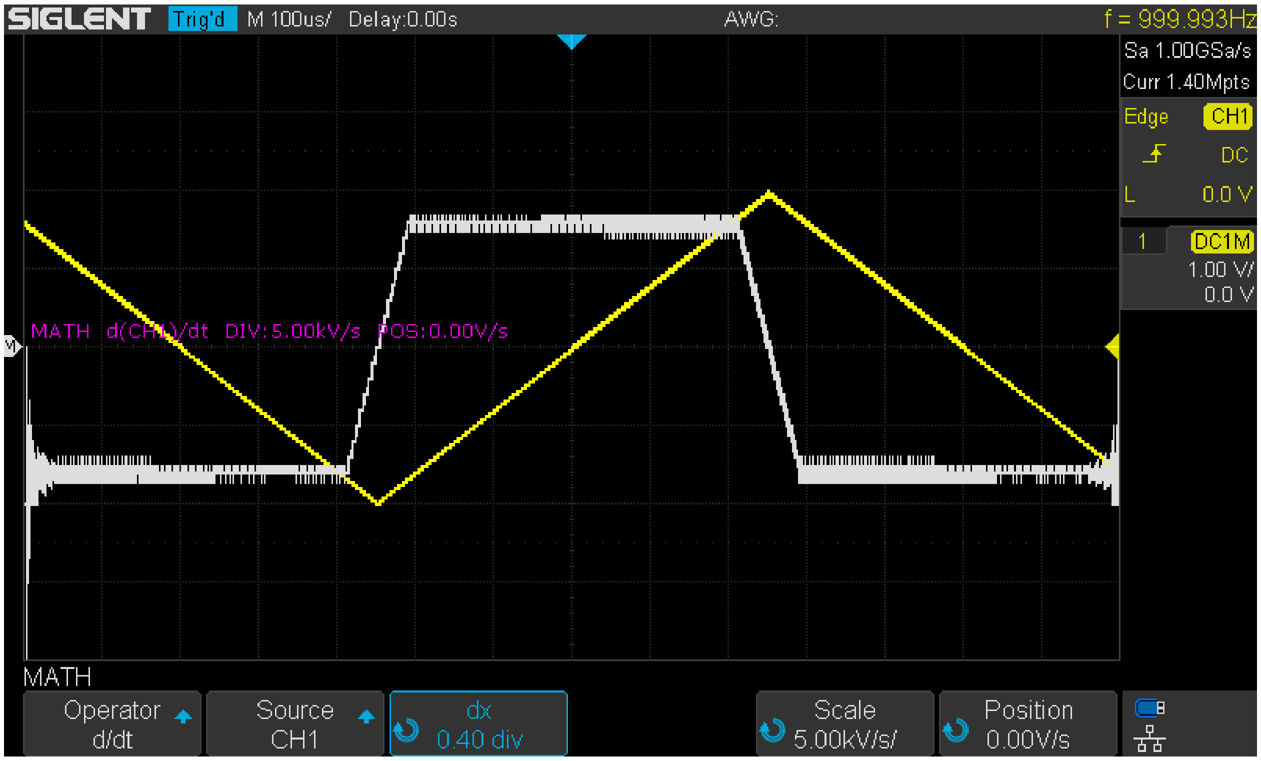


SDS1104X-E_Dif_Ramp_Avg16_2%



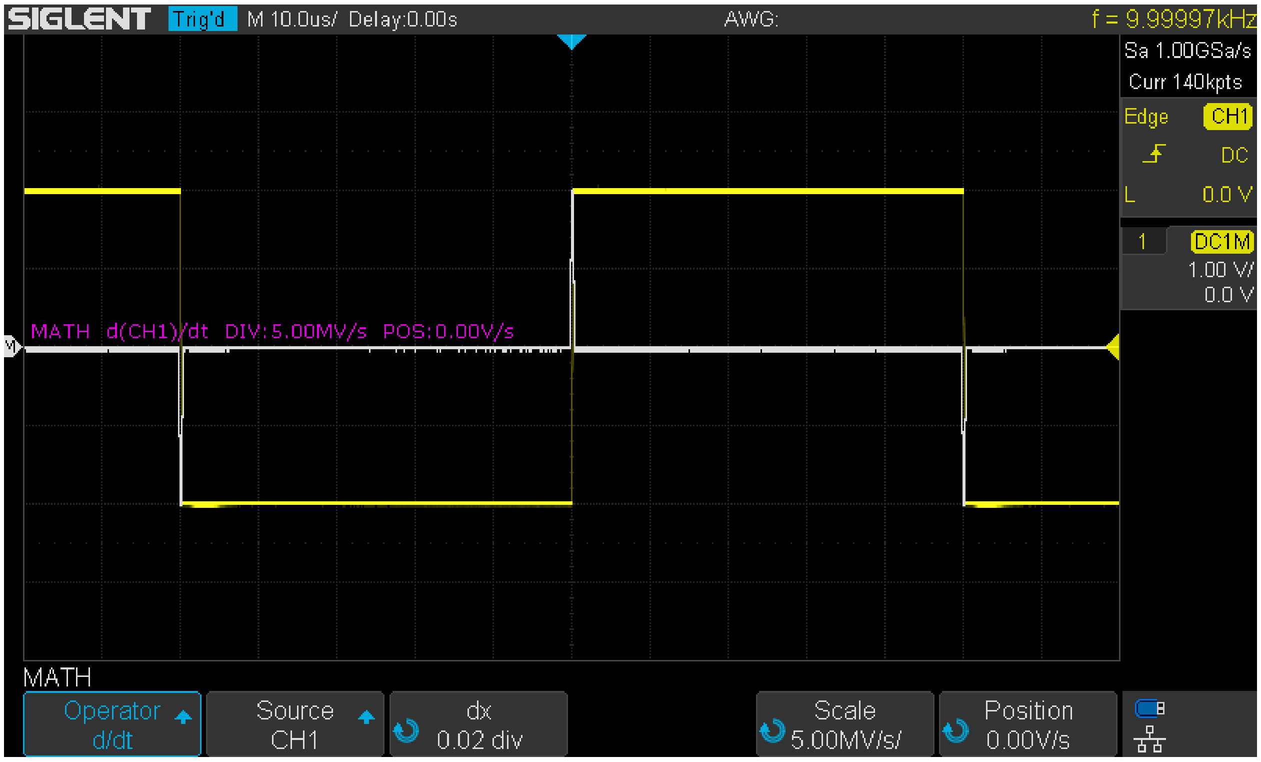
SDS1104X-E_Dif_Ramp_Avg16_4%

Even though the noise gets better with increasing time intervals, the results are still not very usable. With the maximum value 0.4 for *dx,* we get very slow transitions for the resulting rectangle, yet the math trace is fat and noisy.



SDS1104X-E_Dif_Ramp_Avg16_40%

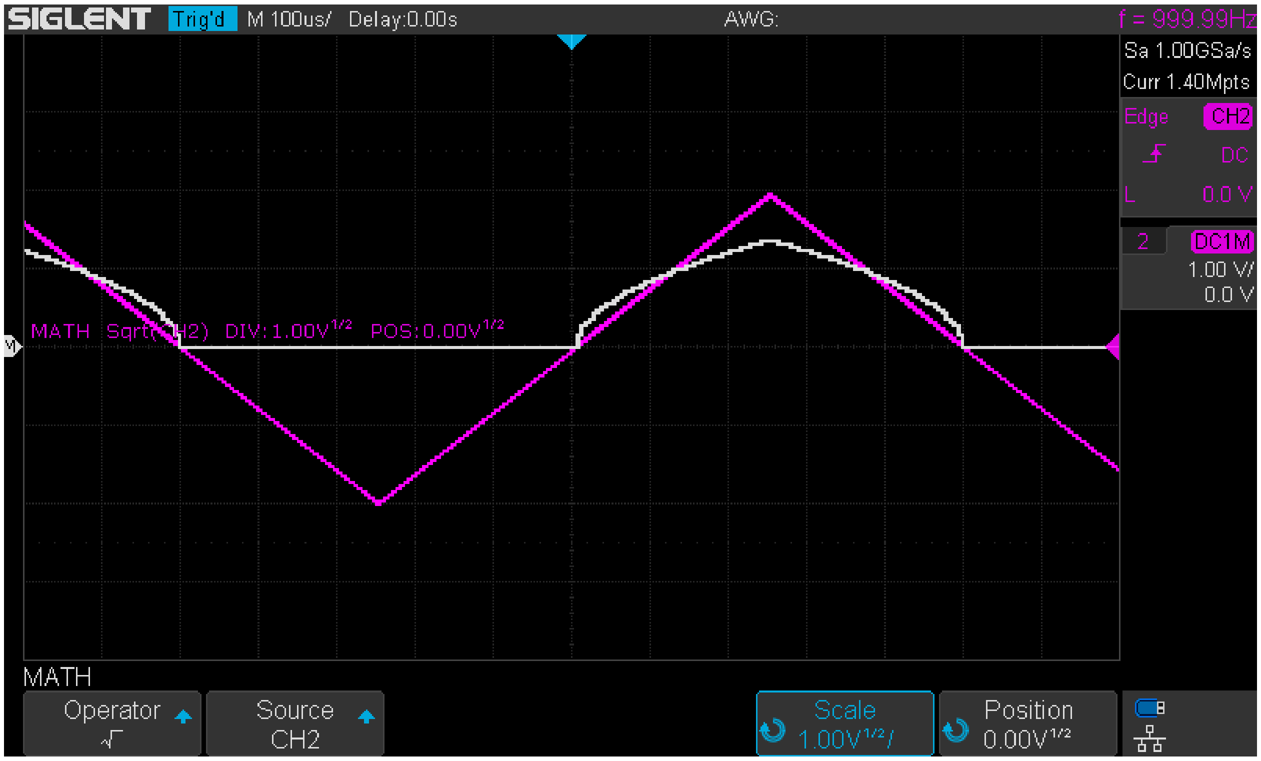
A square wave with fast transitions on the other hand is a perfect candidate for the differentiate function, and particularly so with the lowest possible *dx* parameter of 0.02.



SDS1104X-E_Dif_Square_2%

Square Root

Might be useful to convert some sensor signal, that represents power, back into voltage/current.



SDS1104X-E_Sqrt

Poor Men’s Differential Probing

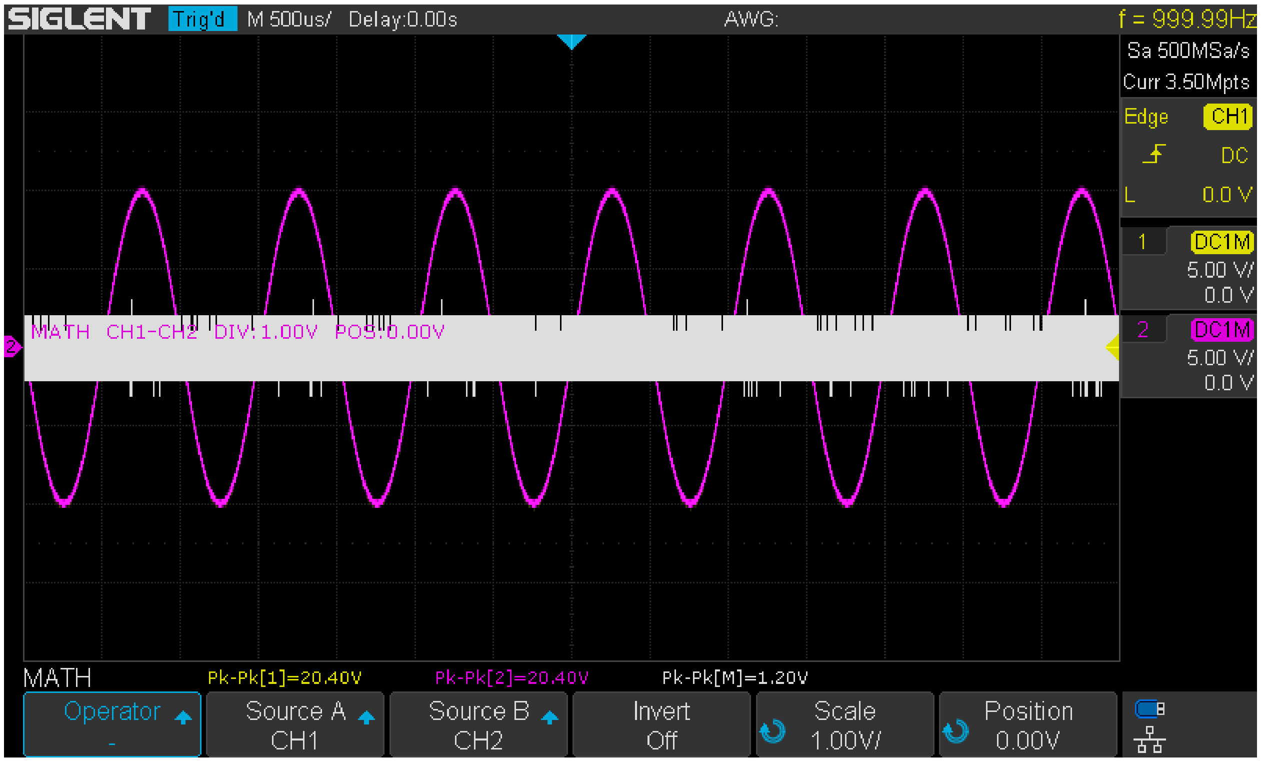
With analog scopes, we were able to combine two regular (single ended, ground referenced) channels into one differential channel. This was done by adding both channels with the 2nd channel inverted, whose gain had to be fine tuned in order to give a maximum of common mode rejection. Of course, this solution was far from ideal and sensitivity as well as common mode rejection were rather limited, especially at higher frequencies, which made it hard to get meaningful results when common mode voltages were high compared to the differential signal.

Digital scopes generally seem to have pretty much lost that functionality, even though the preconditions appear good at first glance. After all we have a separate math channel that has its own individual gain and position, which should be ideal for viewing small signals in presence of large common mode voltages. But the problems start already when we attempt to fine-adjust the channel gain in order to cancel out the common mode signal as much as possible; this is just not possible. In fact, the math functions completely ignore the channel gain settings except for the input attenuator, which just cannot be ignored. This seems to be convenient at first, as it makes the math channel somewhat independent from the regular channel settings, but makes it impossible to fine-tune the gain in order to get it equal across the channels.

Furthermore, the individual gain for the math channel doesn’t actually help, as it can only be a software zoom and an 8-bit resolution is not up to the task, particularly when the math gain is set higher than the channel gain.

The screenshot below demonstrates the result of two identical signals fed into channels 1 and 2 at 5V/div and a difference math channel is set to a 5 times higher gain at 1V/div. The result is just an approximately 1 division wide bar. Common mode rejection can be estimated from the amplitude measurements and would be 1.2V/20.4V = 0.0588 ~ -24.6dB, which is certainly not sufficient for differential measurements.

As a conclusion, poor men's differential probing doesn't work with this scope and the same is most likely true for the majority of other DSOs as well. For tasks that require floating measurements and/or differential probing, there really is no way around getting the appropriate physical probes.



SDS1104X-E_PoorMen_Diff_Probing

FFT

Now we've finally arrived at one of the highlights of the SDS1kX-E series, the 1Mpts Fast Fourier Transform, which turns the scope into some sort of spectrum analyzer. This can be found in the math *Operator* menu, yet it deserves a dedicated chapter simply because it's almost something like an instrument within the instrument.

Thankfully, Siglent have implemented a more spectrum analyzer oriented user interface instead of treating the spectrum plot just like an ordinary Y-t signal trace. This means a big plus in terms of usability.

Even though there are basically no automatic measurements, markers or analysis applications, which is in contrast to modern spectrum analyzers and higher end DSOs, this does not prevent us from getting all the answers we could possibly expect when using an 8-bit DSO for analyzing a signal in the frequency domain. We should not forget that there have been times where even dedicated SAs had barely anything but a manual marker to determine the approximate frequency at any point of the spectrum – and yet engineers got their job done.

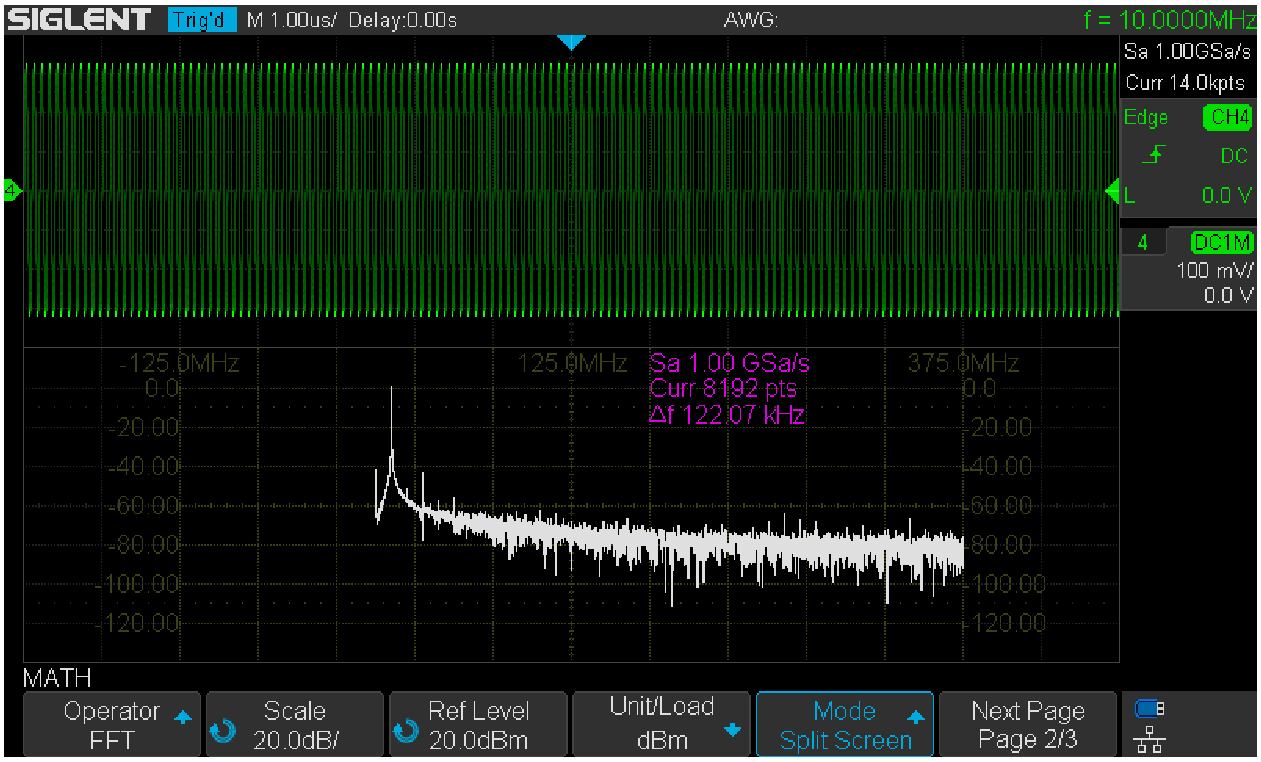
Generally speaking, we should not get too obsessed with features and gadgets, but concentrate more on the quality of measurements. All the bells and whistles won't serve us anyway, unless we get low distortion and noise, accurate levels, a reasonable high spurious-free dynamic range and adequate frequency resolution. These are parameters that really matter for any SA and make the difference between a useful tool or just another "me too" feature.

Because of this, the FFT will be evaluated just like the base functionality on any real spectrum analyzer, so this is going to be a rather lengthy review.

Basic Operation

This section covers some of the fundamentals of the user interface that need to be understood as a basis for the following chapters.

FFT is enabled by pressing the [**Math**] button on the front panel and then selecting “FFT” from the *Operator* menu. This will show the FFT window with some random settings and hopefully in Split Screen mode. If not, go to page 2 of the *FFT* menu and set this mode there.



FFT_first_open

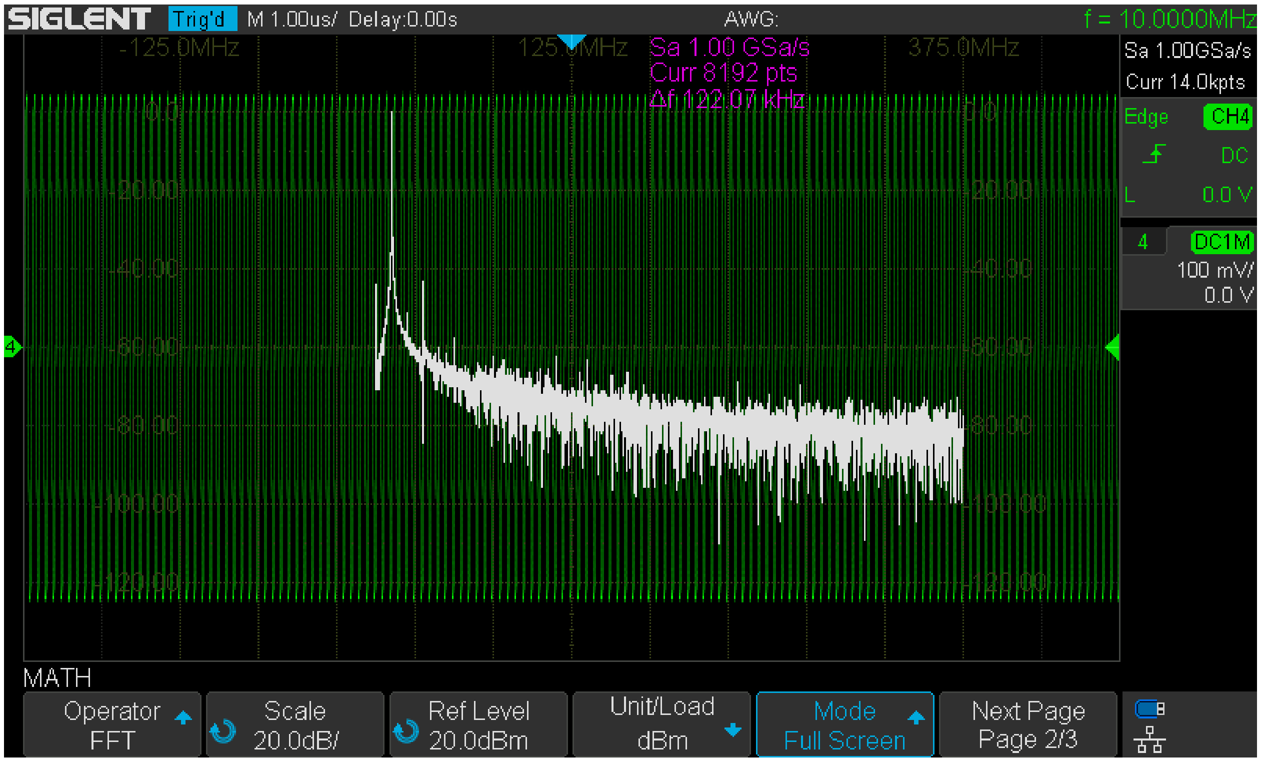
Screen Modes

This is the very first thing to know; there are three different ways to display the FFT:

- Split Screen
- Full Screen
- Exclusive

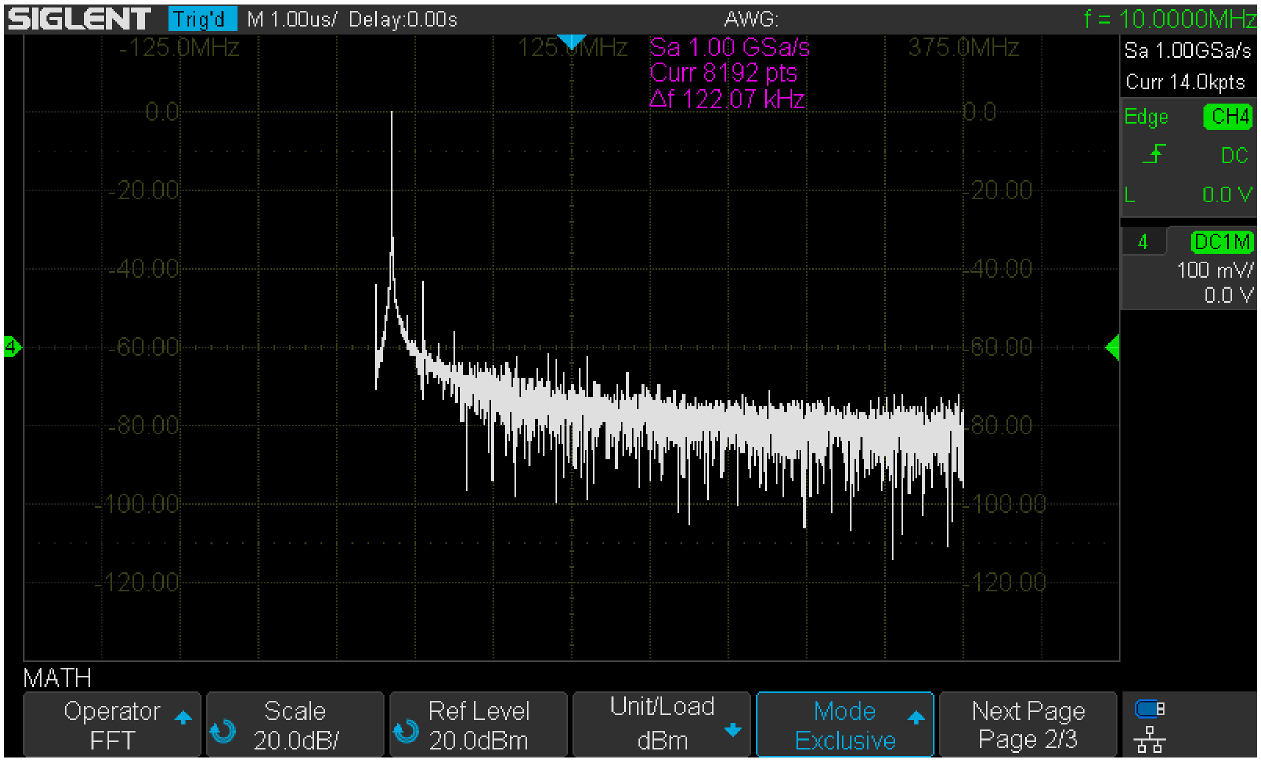
Split Screen as has been shown in the previous screenshot is most useful for setting up the measurement, as it allows the observation of the input signal in the time- and frequency-domain in parallel. This is important, because as a general rule we want the signal to nearly fill the screen height in order to maximize the dynamic range. On the other hand, we need to avoid clipping of the input signal and saturating the ADC by all means, because this will lead to distortions and in turn generate lots of unwanted harmonics and spurious signals. Consequently, the previous screenshot shows a proper setup of the vertical gain and/or output of the signal source and this has to be observed during setup.

Full Screen could serve the same purpose as split screen, but now both frequency and time domain are stacked on top of each other which obscures the grid and its labels and might be distracting in general. This is also why I personally don’t like this mode very much and rather use Split Screen whenever I need to see the Y-t display.



FFT_Mode_Full_Screen

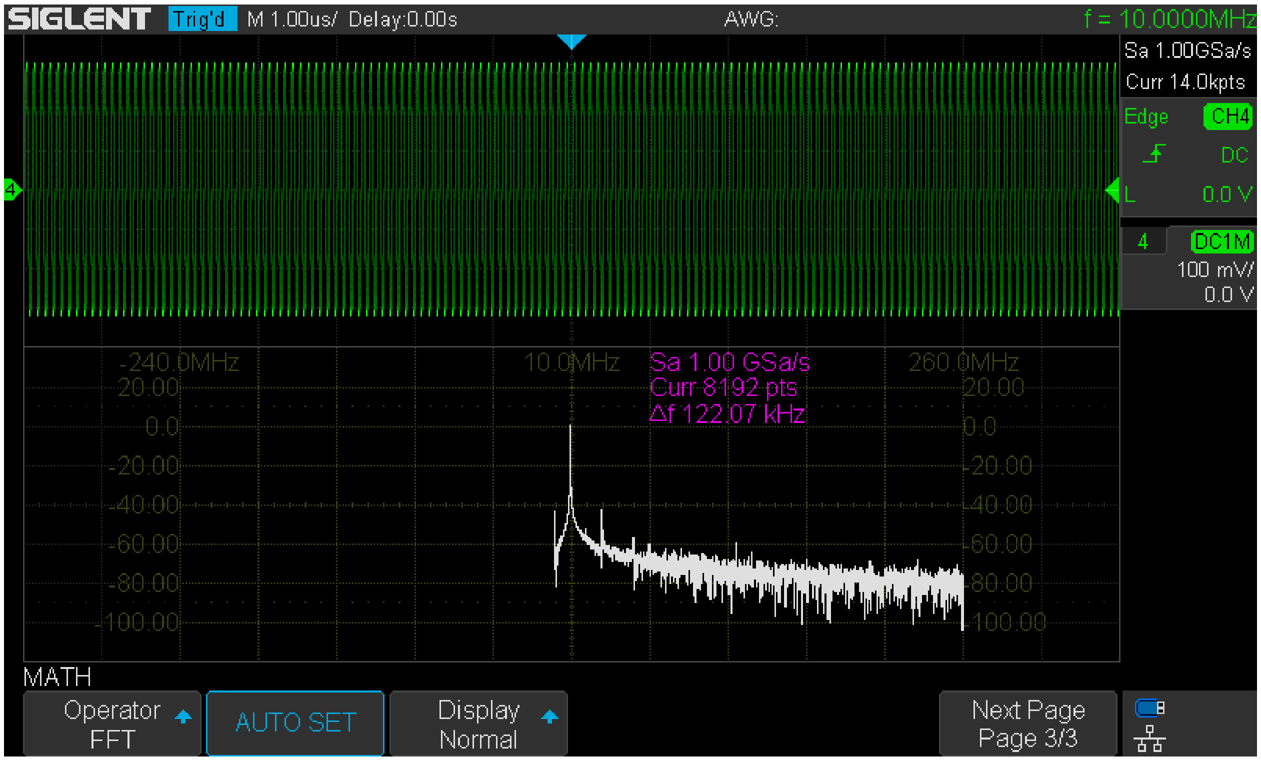
Exclusive shows the FFT exclusively and makes the instrument look more like a dedicated spectrum analyzer. Use this mode once the measurement is properly set up and the input signal amplitude is more or less stationary.



FFT_Mode_Exclusive

AUTO SET

This can be found on page 3 of the *FFT* menu and generally does a lousy job. It sets the center frequency according to the strongest signal, thus could be used as a *starting point* for narrowband signal analysis. Other than that, it's best to stay away from the AUTO SET function and there is no way to avoid setting up the instrument manually as described later in this document.



FFT_Auto_Set

Units

This submenu can be found on page 2 of the *FFT* menu. The units used for amplitude measurements, i.e. the labeling for the Y-axis of the grid, are selected here. The same submenu also allows the specification of the external load impedance. This is required for the dBm units, which specify a power level instead of voltage/current.

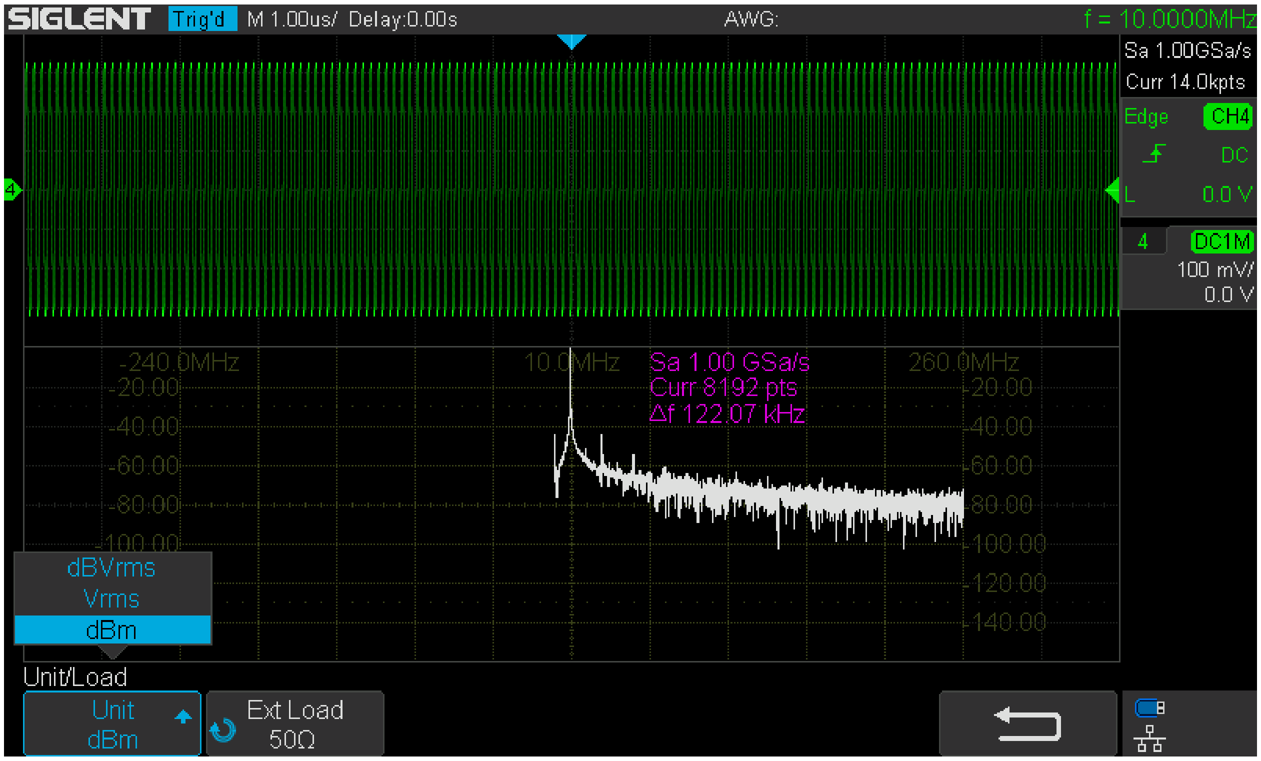
For general laboratory use, we would universally have an external 50Ω pass-through terminator, hence a 50Ω load impedance. The same is true for most RF applications, even though there are exceptions especially for the antenna inputs of domestic broadcast receivers. 75Ω is the standard for video signals, as well as 600Ω for professional Audio. For Audio in general, the use of dBm is not common but might be useful for characterizing the output of power amplifiers, where it comes in handy that the Siglent FFT allows us to specify load impedances down to 1Ω.

dBVrms is the relative voltage level expressed in decibel with reference to 1Vrms. This is the right setting for (unspecified) high impedance source/load configurations.

Vrms is the absolute voltage, which also means that we get a linear Y-axis that severely limits the dynamic range that can be displayed. I really don't see much use for that.

dBm is the relative power level expressed in decibel with reference to 1mW into the specified load impedance. This is the preferred unit for general laboratory use as well as RF applications.

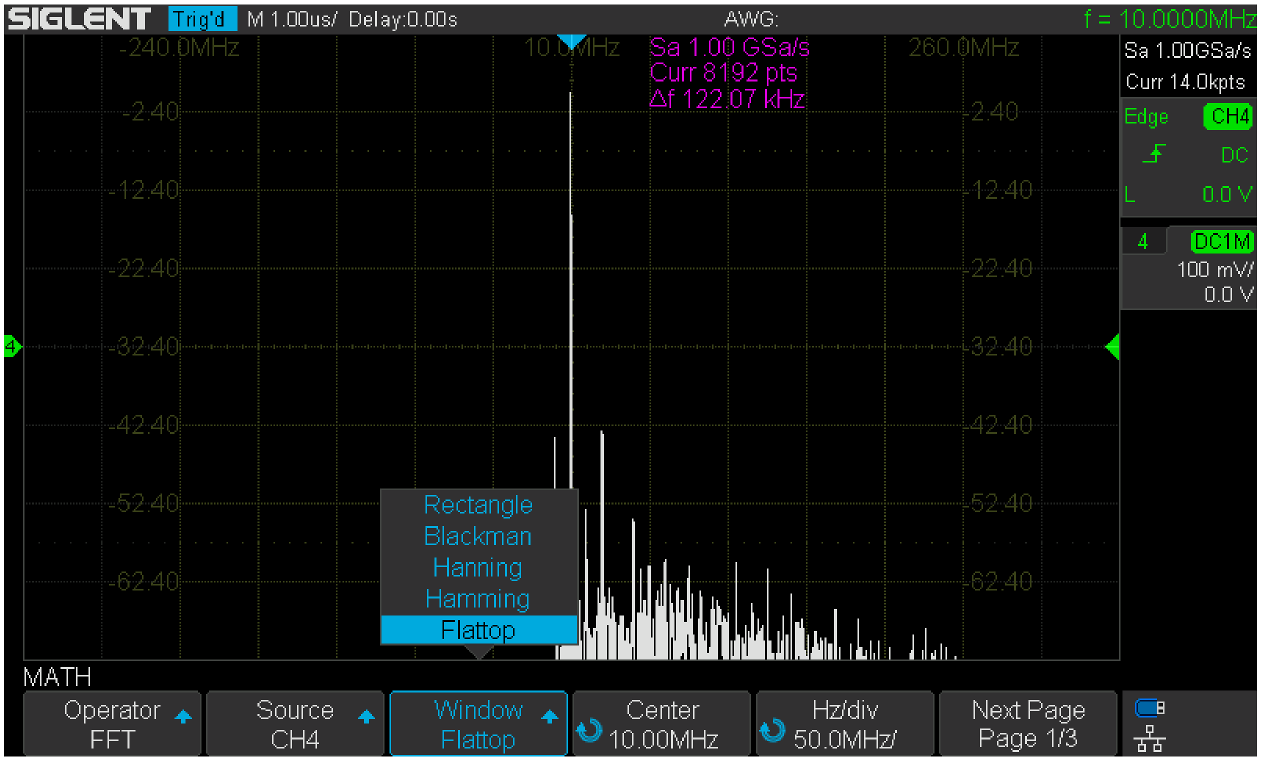
The screenshot below shows how the FFT is set up for dBm units and 50Ω external load. At the same time, an external 50Ω pass-through terminator has been fitted for channel 4.



FFT_Units

Window

Another important setting is the window function on page 1 of the *FFT* menu.



FFT_Windows

This is roughly equivalent to selecting the properties of the final IF filters in a real SA. Of course, the latter do not offer such a choice, apart from the selectable IF bandwidth. But FFT works differently than a swept spectrum analyzer; we cannot set the analysis bandwidth directly and the “final IF filter” is just some signal processing on a limited set of sample data (record) gathered within a certain time interval, where the actual input signal is not zero outside this interval. This causes computation errors and produces artifacts. This is why the window function is required to do some pre-conditioning on the sample data and it comes down to providing a suitable compromise between bandwidth, amplitude accuracy, filter shape and selectivity including leakage (side lobe suppression). The simplest window is the rectangle which has the narrowest -3dB bandwidth but very poor properties otherwise. Historically a number of window functions exist that provide a better compromise at the expense of an increased -3dB bandwidth. Some window functions have been particularly optimized for a certain property and these are also the most beneficial ones for practical use – in my book, at least.

The table below gives an overview of the window functions available in the SDS1104X-E. Note that some windows have parameters (Hanning in this selection) and I do not know what Siglent has actually implemented, so the table can only show a span of properties for the usual range of these parameters. Also note that “Bins” refers to what is displayed as “Δf” (frequency step) within the FFT window.

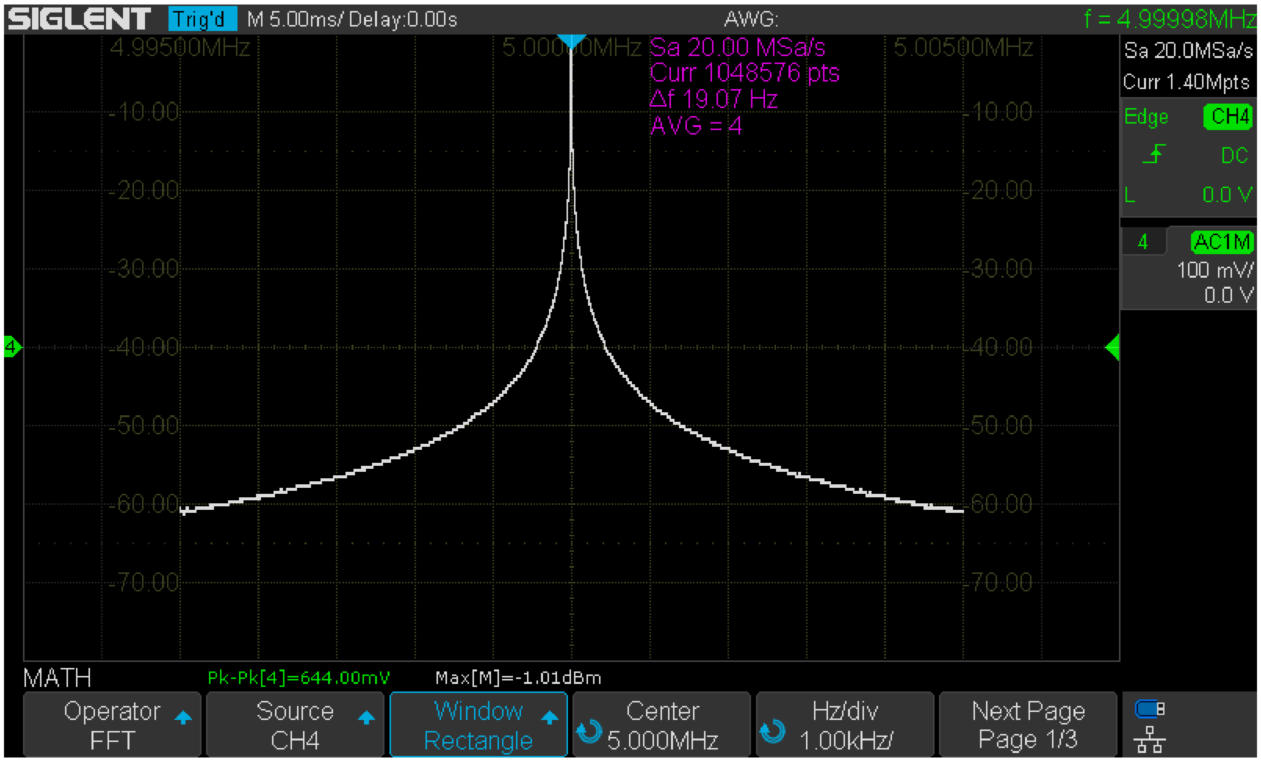
Window	**-3dB BW (Bins)**	**Max. Side Lobe [dB]**	**Side Lobe roll-off [dB/Oct.]**	**Remark**
Rectangle	0.89	-13.2	6	Min. 3dB bandwidth, used for short transients
Blackman	1.68	-58	18	High side lobe suppression, used for audio
Hanning	1.20 ~ 1.86	-23 ~ -47	12 ~ 30	Used for audio and vibration measurement
Hamming	1.30	-41.9	6	Used for speech analysis
Flattop	2.94	-44	6	Negligible pass band ripple, used in spectrum analyzers and for calibration

Many more window functions do exist and I personally would love to have Gaussian and particularly Blackman-Harris available, but that’s not really a problem, especially not for an 8-bit system. For the time being I have the following recommendation for anyone who just wants to use the FFT right away without striving for an university degree in digital signal processing (RBW = Resolution Bandwidth):

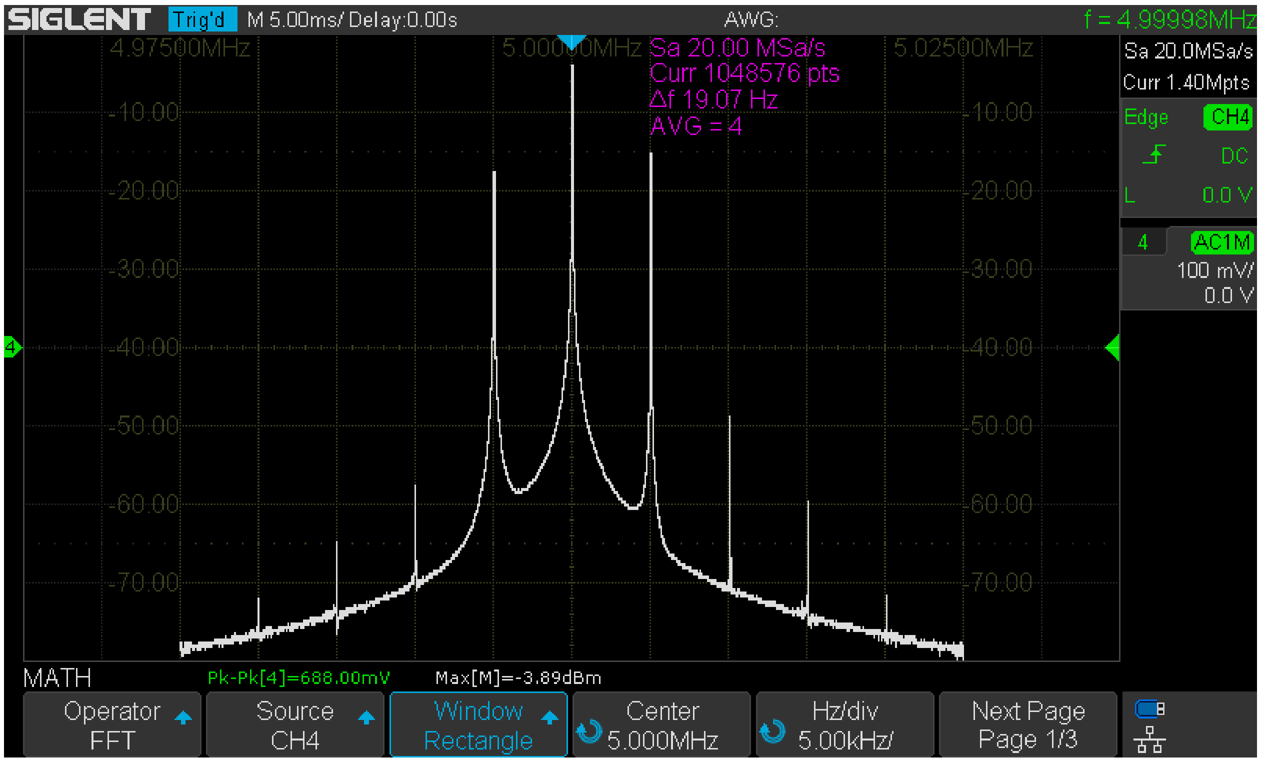
- Use Flat-Top whenever best amplitude accuracy is important. RBW ~ 3 x Δf
- Use Blackman otherwise, especially when a high dynamic range is desired. RBW ~ 1.7 x Δf

The following screenshots show all available window functions in two situations:

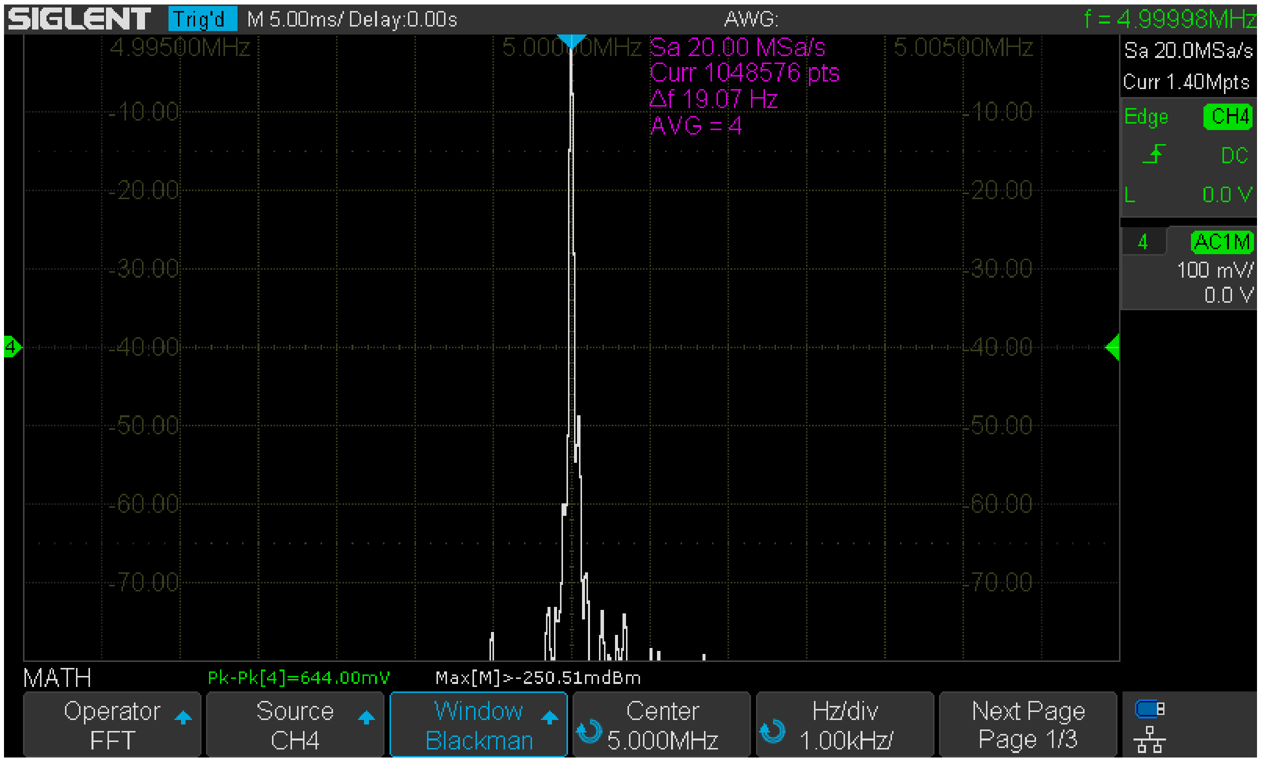
1. Sine, 5MHz, 0dBm, displayed at 1kHz/div to show the selectivity and filter shape
2. Sine, 5MHz, -3dBm and 50% AM with 5kHz, displayed at 5kHz/div



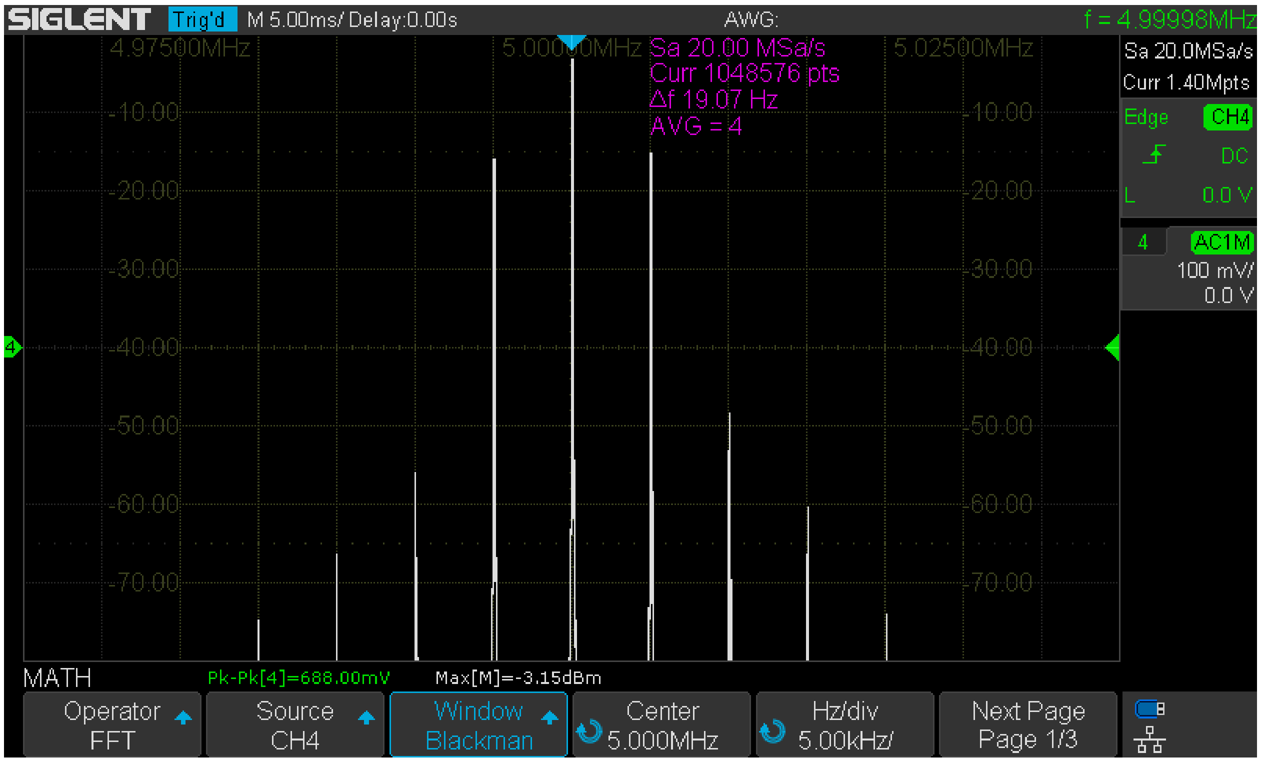
SDS1104X-E_FFT_Rectangle



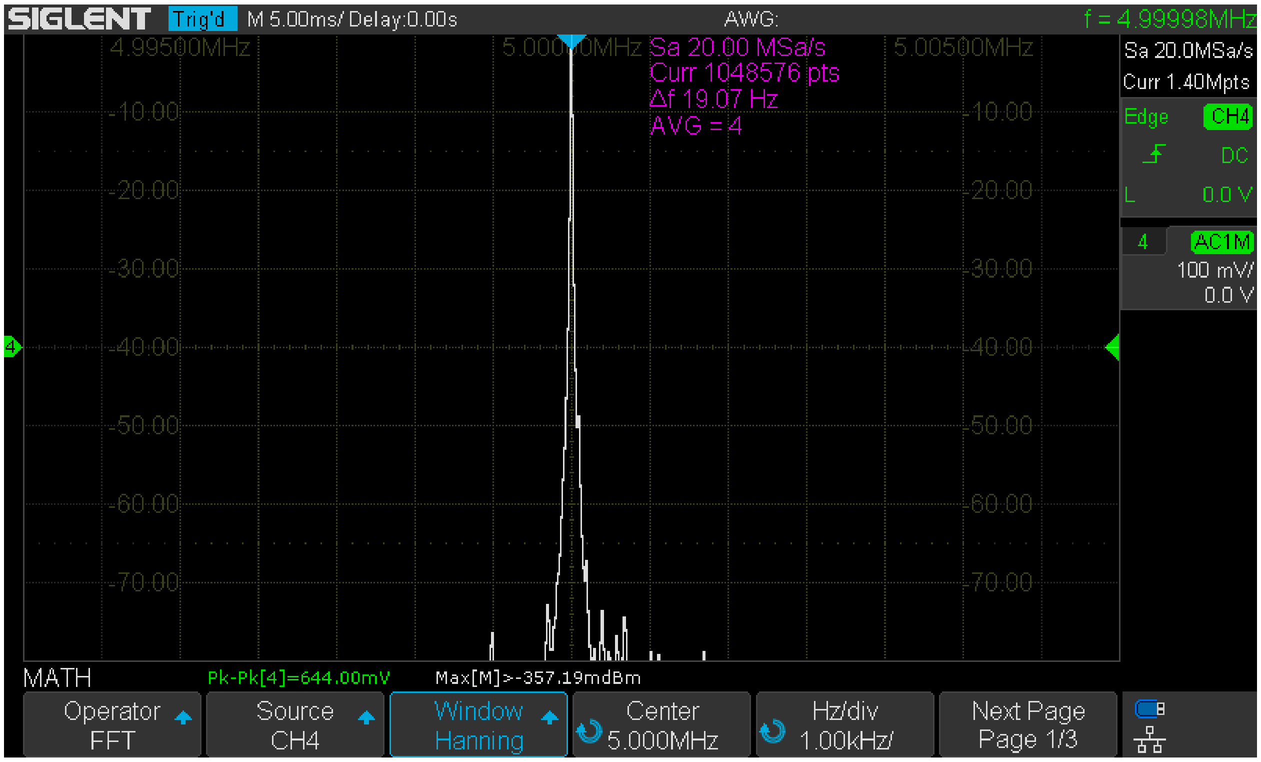
SDS1104X-E_FFT_Mod_Rectangle



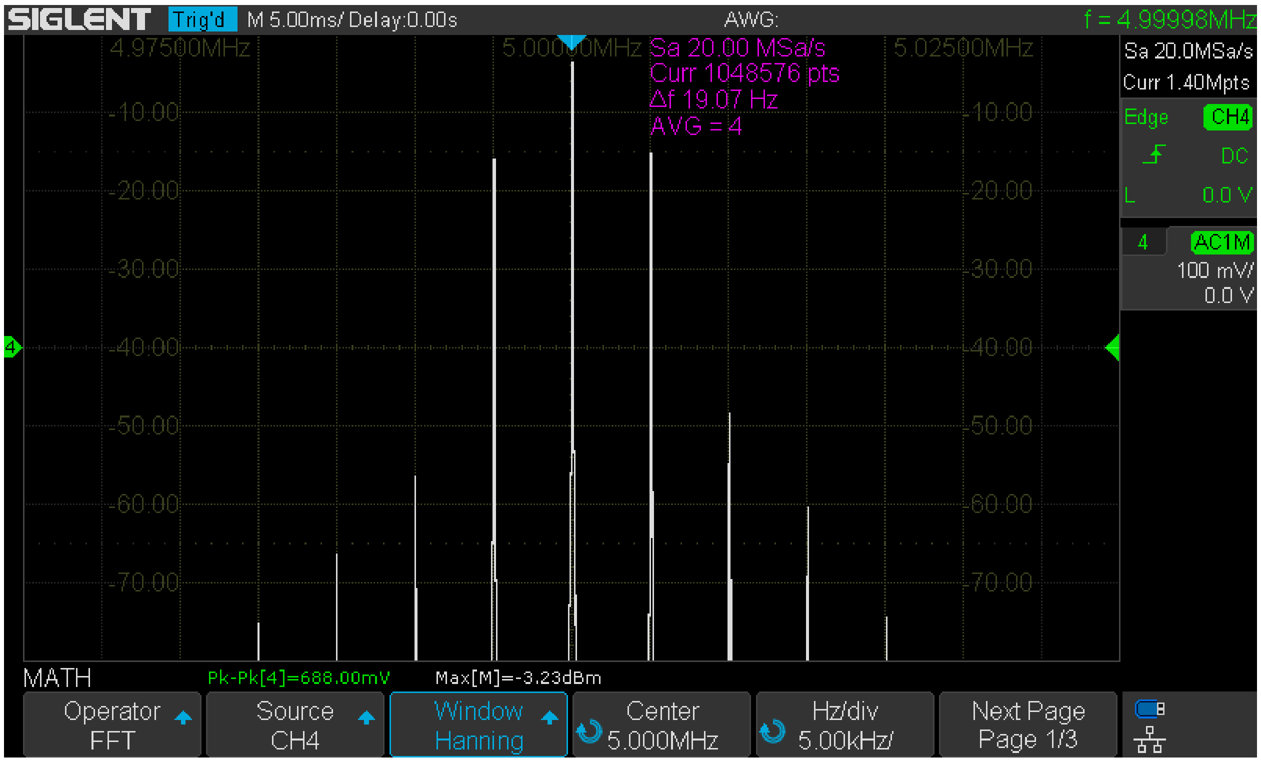
SDS1104X-E_FFT_Blackman



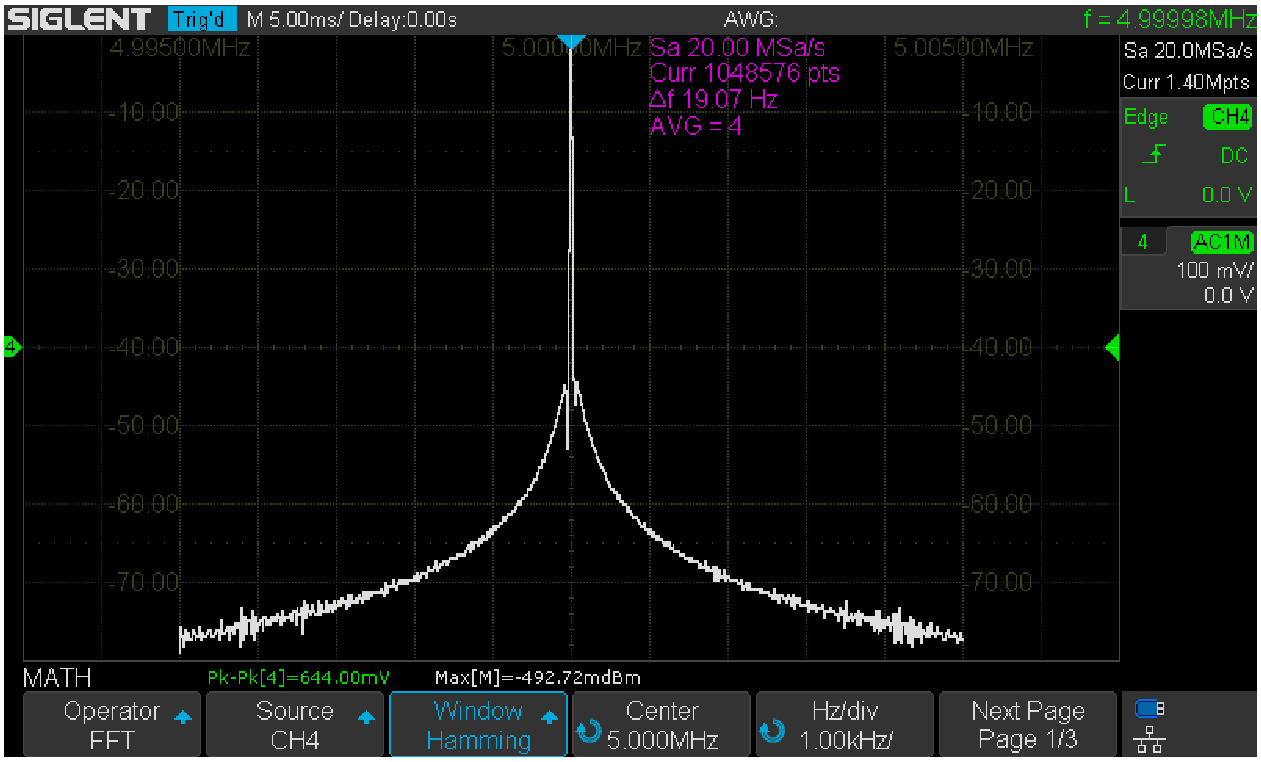
SDS1104X-E_FFT_Mod_Blackman



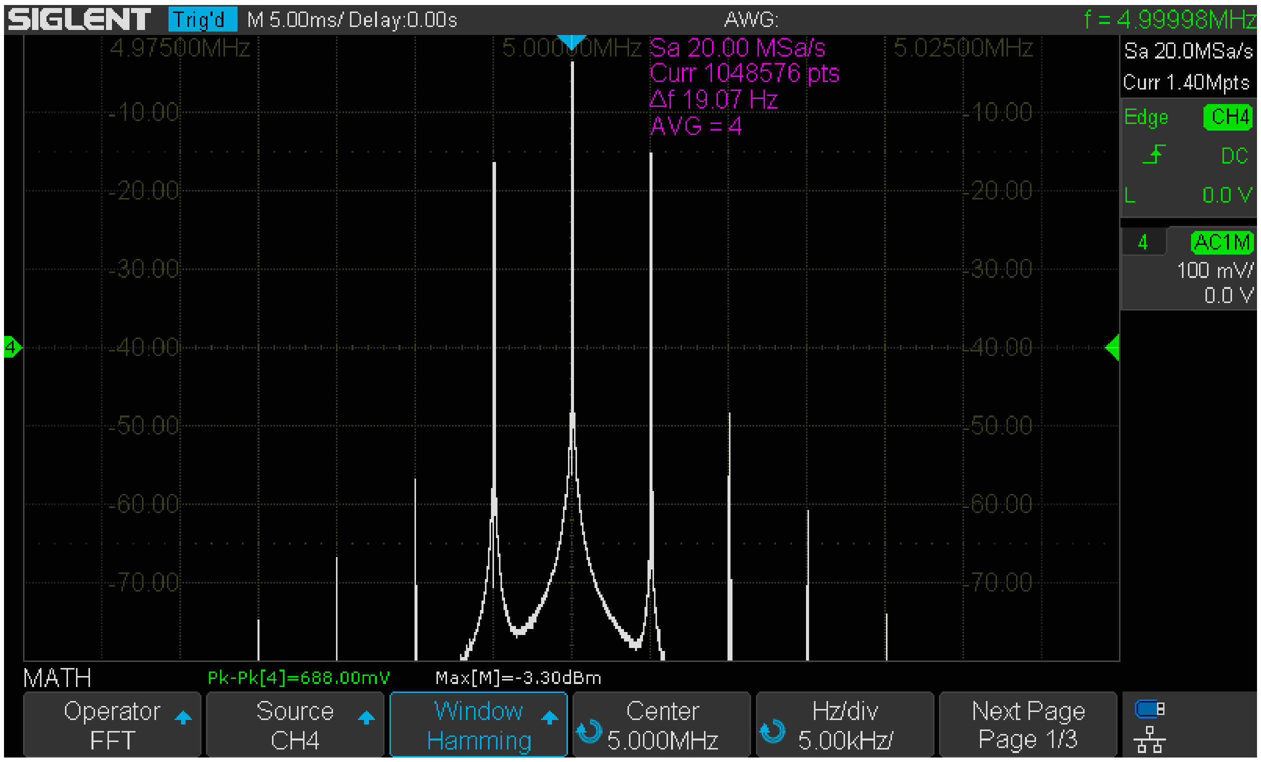
SDS1104X-E_FFT_Hanning



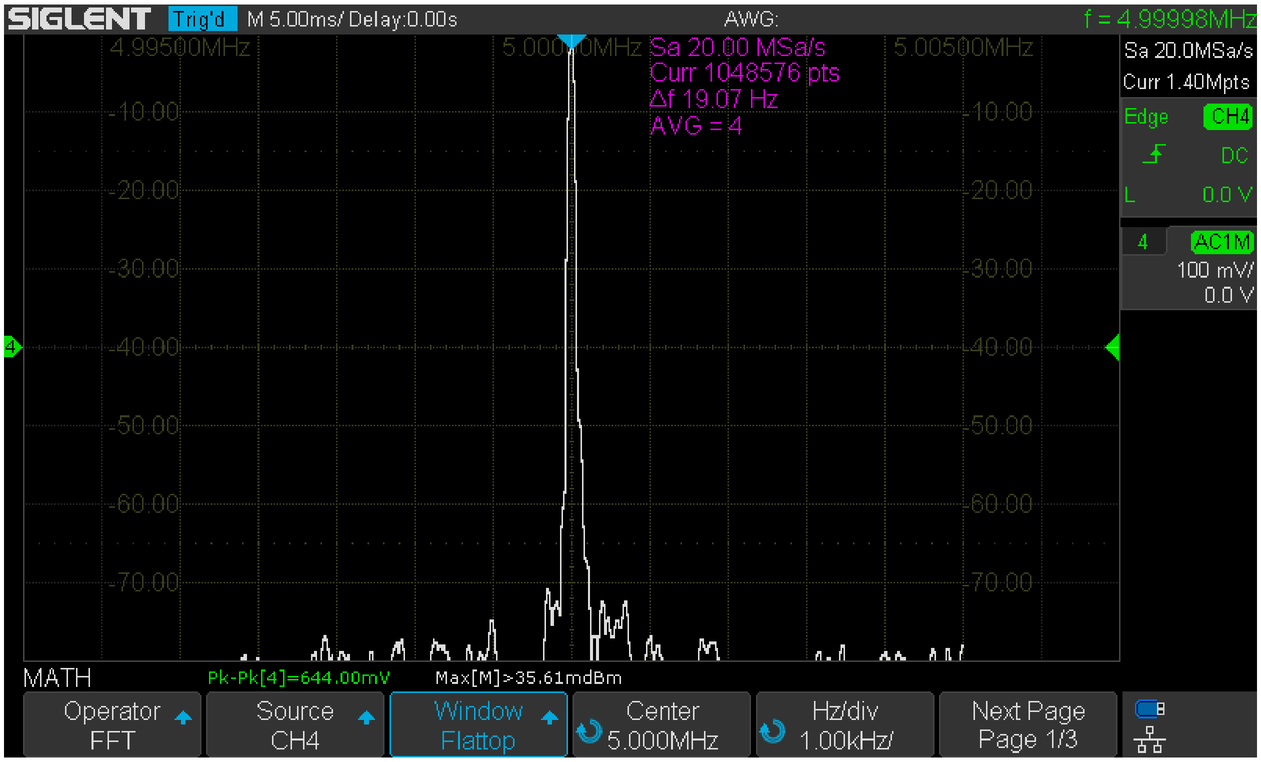
SDS1104X-E_FFT_Mod_Hanning



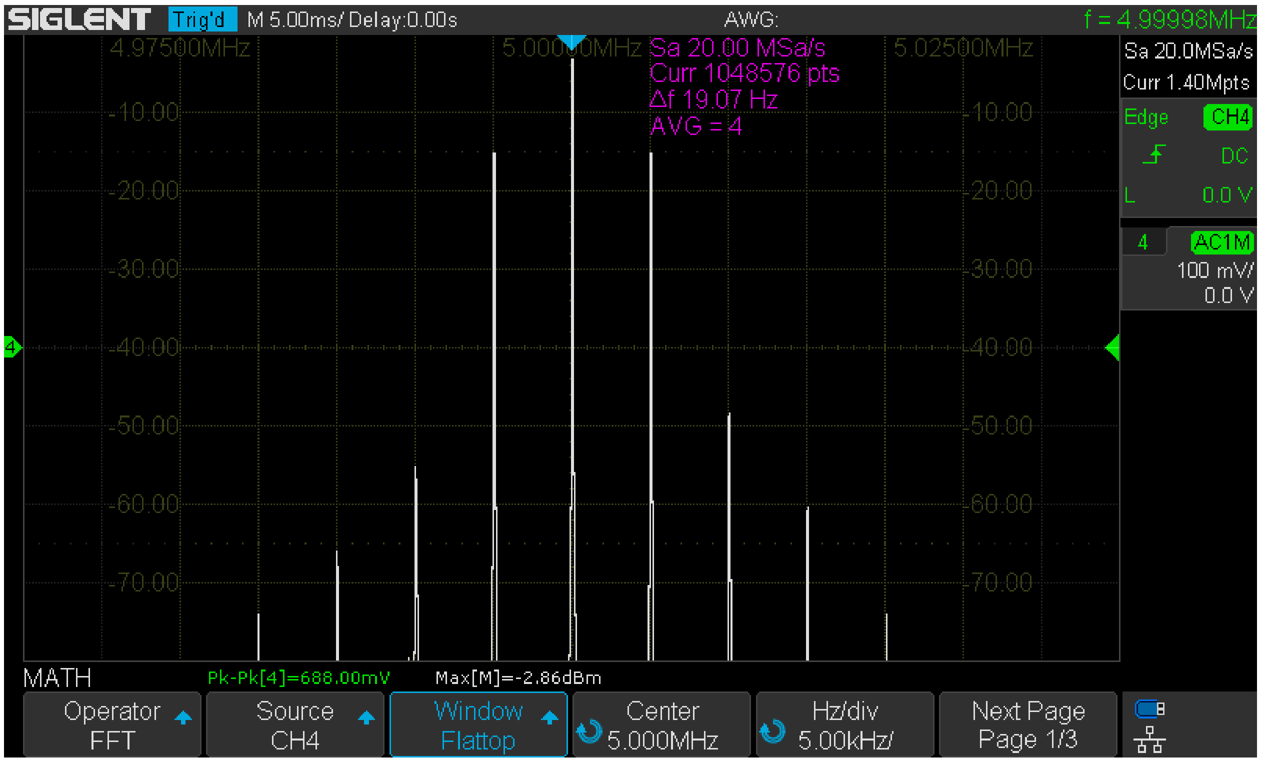
SDS1104X-E_FFT_Hamming



SDS1104X-E_FFT_Mod_Hamming



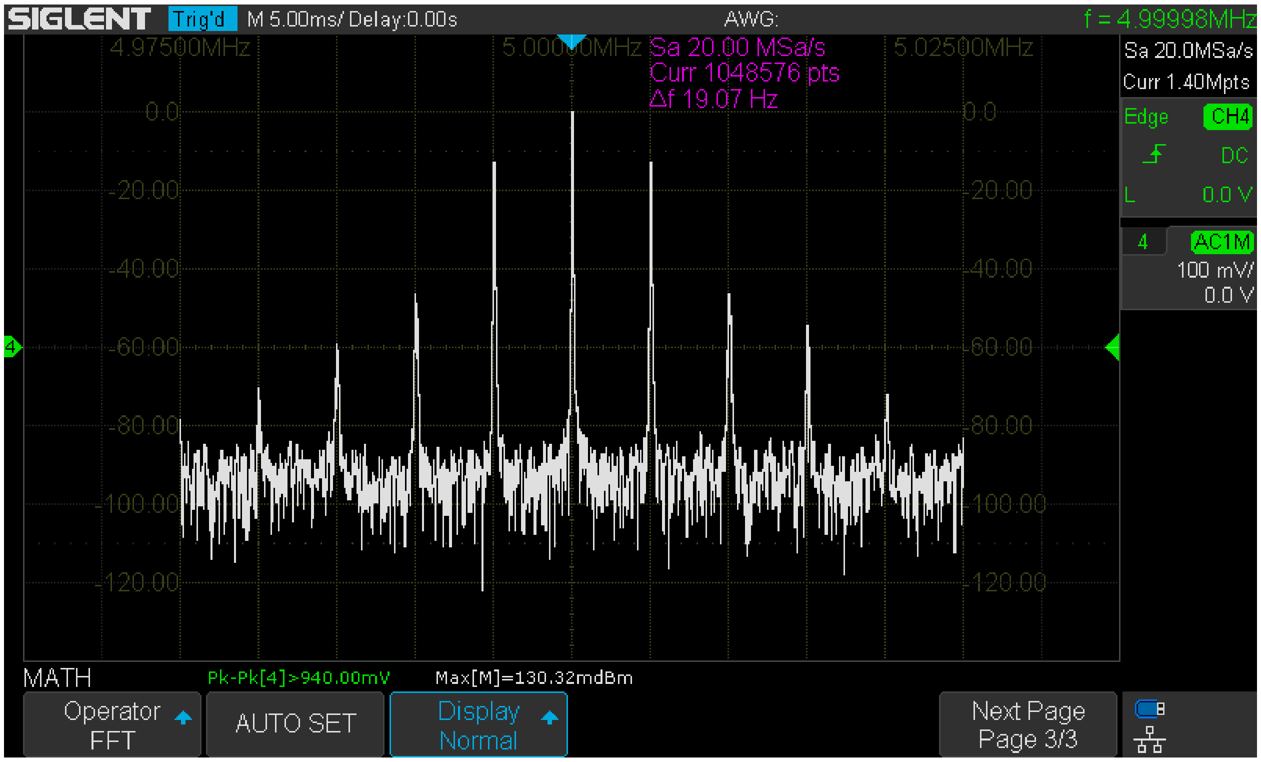
SDS1104X-E_FFT_Flattop



SDS1104X-E_FFT_Mod_Flattop

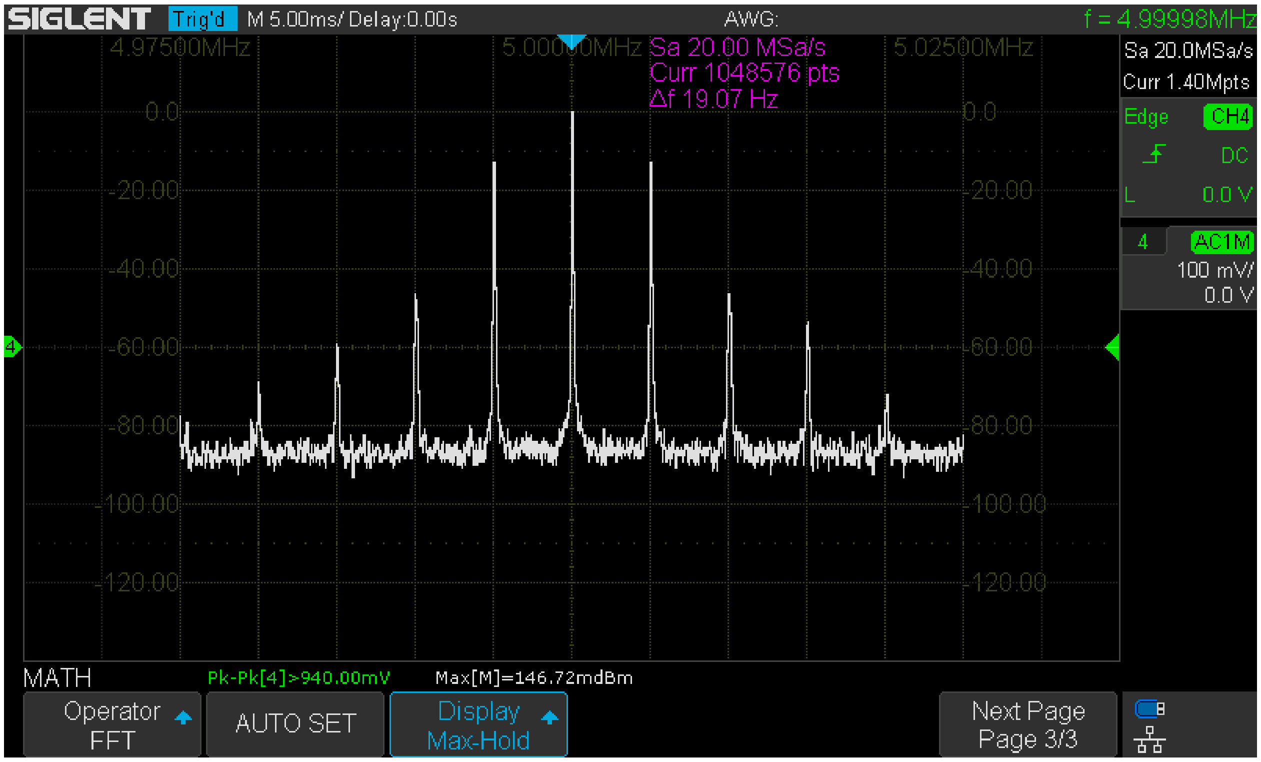
Display Modes

Normal mode will reflect any input signal change instantly and completely on the following FFT trace.



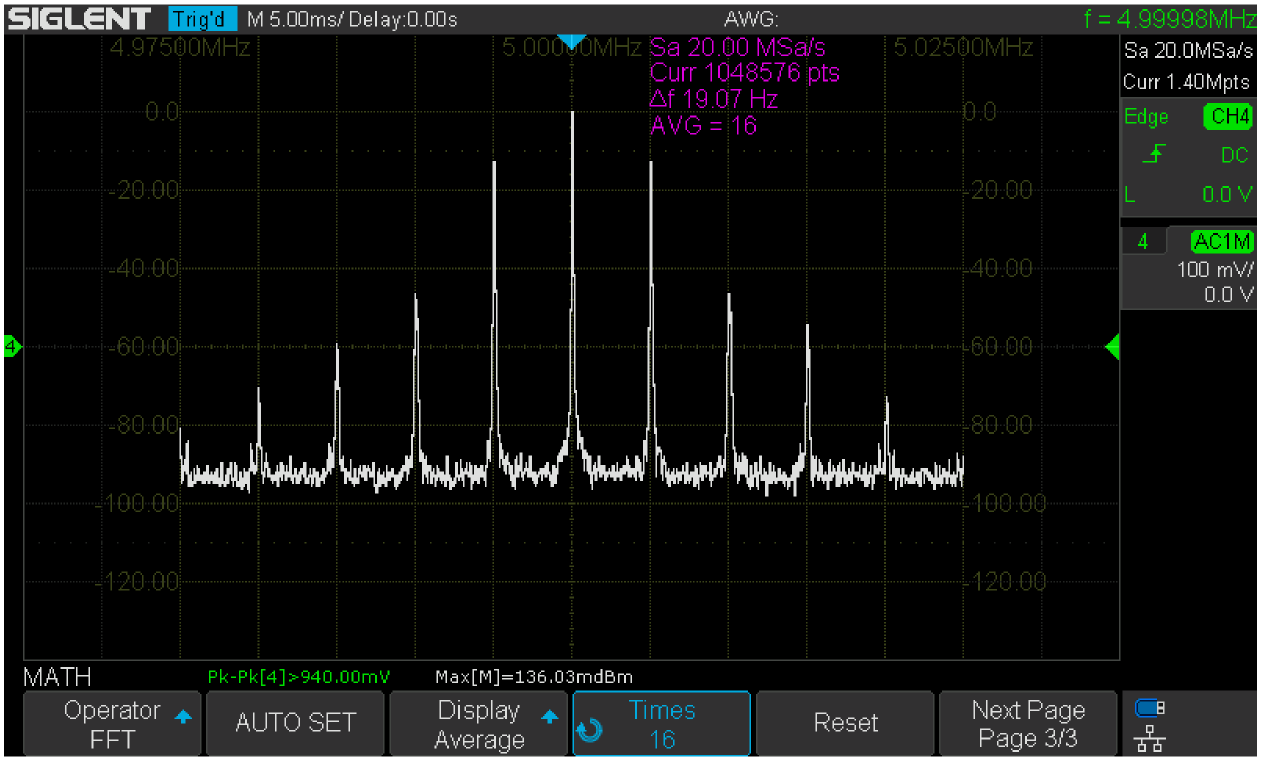
SDS1104X-E_FFT_Mode_Normal

Max-Hold only keeps the long-term maximum values for each frequency bin.



SDS1104X-E_FFT_Mode_Peak_Hold

Average builds the sliding average over the number of records specified by the *Times* soft menu item, which can be set to any value between 4 and 1024.



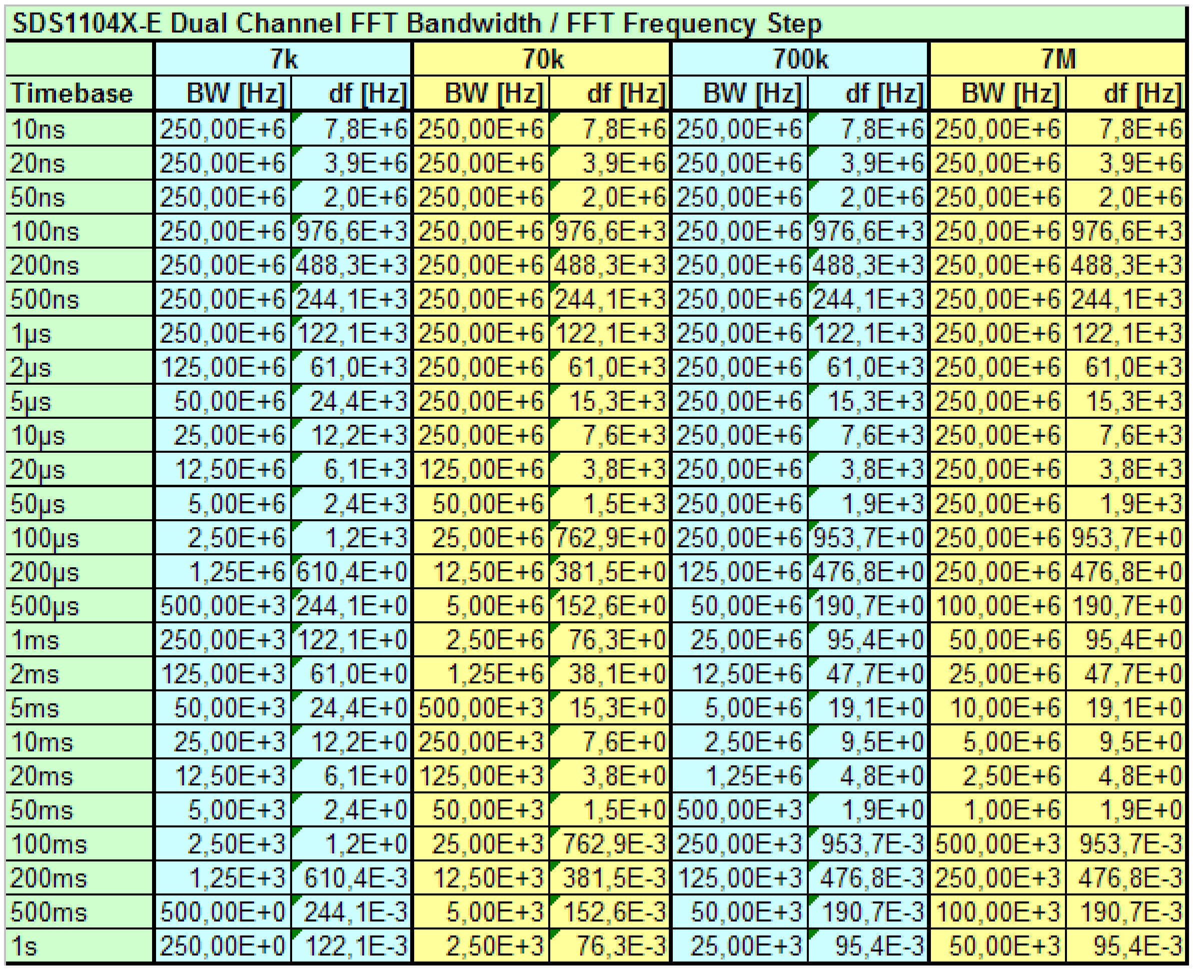
SDS1104X-E_FFT_Mode_Average16

FFT-Bandwidth and RBW

This is quite different to a real SA. There is no menu for the resolution bandwidth and also no direct setting for the FFT-bandwidth, even though we have a soft menu item for the horizontal scale in Hz/div, which ultimately specifies the visible span. But this is just for zooming into a longer FFT trace and for best speed and lowest RBW we need to make sure that no high zoom factor is required to get the display we want. The following rules apply:

- The analysis bandwidth (FFT-BW) is always half the sample rate.
- The frequency step (Δf) is the sample rate divided by the number of FFT points.
- The resolution bandwidth (RBW) is the frequency step multiplied with a factor specific for the window function in use.
- The number of FFT points depends on the record length, which in turn increases with slower timebase settings, but is ultimately limited by the maximum memory set in the *Acquire* menu.

The following table shows the FFT bandwidth (column *BW*) and frequency step (delta frequency, column *df*) for all possible memory depths as specified in the *Acquire* menu and for timebase settings from 10ns/div up to 1s/div for both channels within a channel group enabled. Keep in mind that the frequency step has to be multiplied by the factor specific for the applied window function in order to know the -3dB resolution bandwidth. That factor can be found in the column *-3dB BW* of the table that summarizes the available window functions in the *Window* section.



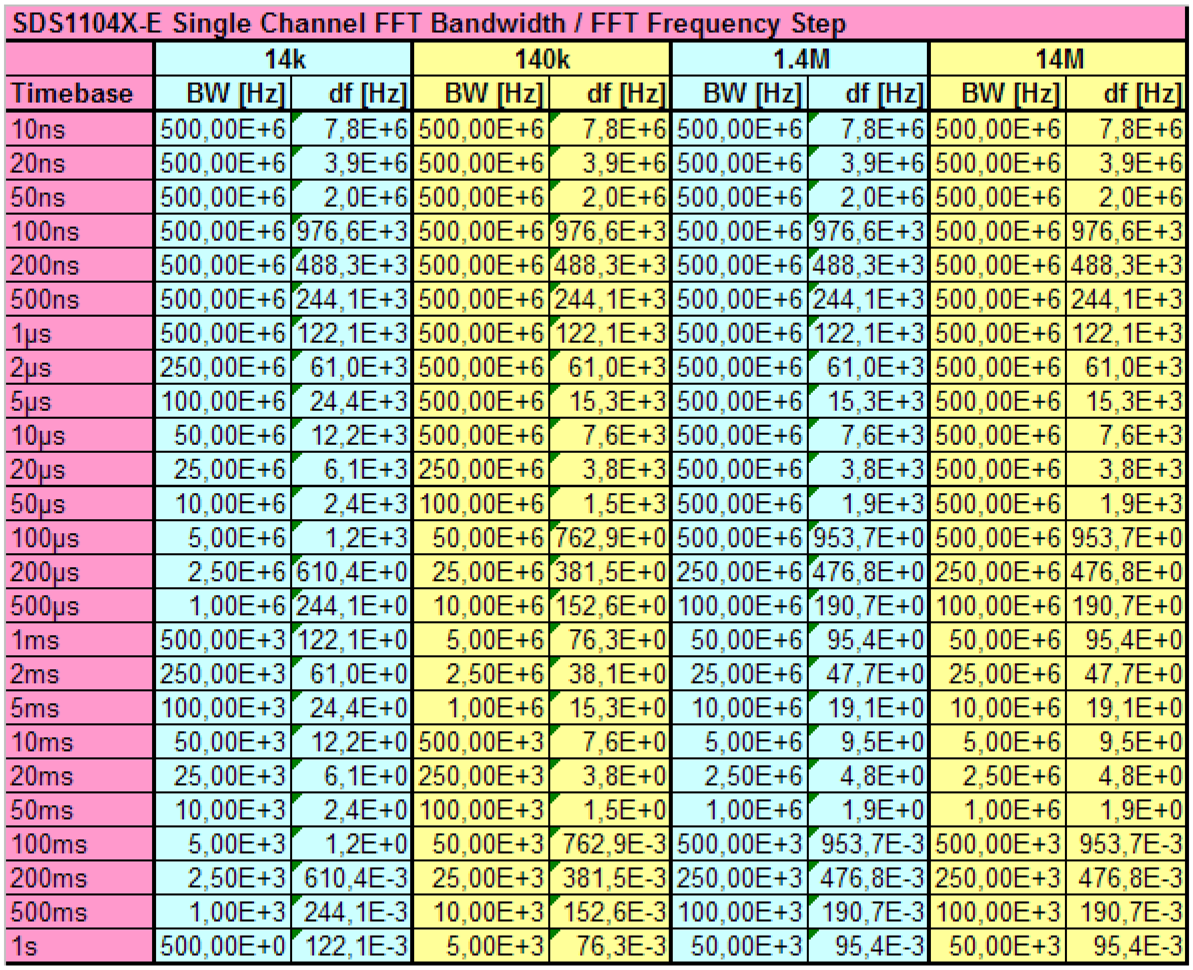
SDS1104X-E Dual Channel FFT Bandwidth / FFT Frequency Step

	7k		70k		700k		7M	
Timebase	**BW [Hz]**	**df [Hz]**	**BW [Hz]**	**df [Hz]**	**BW [Hz]**	**df [Hz]**	**BW [Hz]**	**df [Hz]**
10ns	250,00E+6	7,8E+6	250,00E+6	7,8E+6	250,00E+6	7,8E+6	250,00E+6	7,8E+6
20ns	250,00E+6	3,9E+6	250,00E+6	3,9E+6	250,00E+6	3,9E+6	250,00E+6	3,9E+6
50ns	250,00E+6	2,0E+6	250,00E+6	2,0E+6	250,00E+6	2,0E+6	250,00E+6	2,0E+6
100ns	250,00E+6	976,6E+3	250,00E+6	976,6E+3	250,00E+6	976,6E+3	250,00E+6	976,6E+3
200ns	250,00E+6	488,3E+3	250,00E+6	488,3E+3	250,00E+6	488,3E+3	250,00E+6	488,3E+3
500ns	250,00E+6	244,1E+3	250,00E+6	244,1E+3	250,00E+6	244,1E+3	250,00E+6	244,1E+3
1µs	250,00E+6	122,1E+3	250,00E+6	122,1E+3	250,00E+6	122,1E+3	250,00E+6	122,1E+3
2µs	125,00E+6	61,0E+3	250,00E+6	61,0E+3	250,00E+6	61,0E+3	250,00E+6	61,0E+3
5µs	50,00E+6	24,4E+3	250,00E+6	15,3E+3	250,00E+6	15,3E+3	250,00E+6	15,3E+3
10µs	25,00E+6	12,2E+3	250,00E+6	7,6E+3	250,00E+6	7,6E+3	250,00E+6	7,6E+3
20µs	12,50E+6	6,1E+3	125,00E+6	3,8E+3	250,00E+6	3,8E+3	250,00E+6	3,8E+3
50µs	5,00E+6	2,4E+3	50,00E+6	1,5E+3	250,00E+6	1,9E+3	250,00E+6	1,9E+3
100µs	2,50E+6	1,2E+3	25,00E+6	762,9E+0	250,00E+6	953,7E+0	250,00E+6	953,7E+0
200µs	1,25E+6	610,4E+0	12,50E+6	381,5E+0	125,00E+6	476,8E+0	250,00E+6	476,8E+0
500µs	500,00E+3	244,1E+0	5,00E+6	152,6E+0	50,00E+6	190,7E+0	100,00E+6	190,7E+0
1ms	250,00E+3	122,1E+0	2,50E+6	76,3E+0	25,00E+6	95,4E+0	50,00E+6	95,4E+0
2ms	125,00E+3	61,0E+0	1,25E+6	38,1E+0	12,50E+6	47,7E+0	25,00E+6	47,7E+0
5ms	50,00E+3	24,4E+0	500,00E+3	15,3E+0	5,00E+6	19,1E+0	10,00E+6	19,1E+0
10ms	25,00E+3	12,2E+0	250,00E+3	7,6E+0	2,50E+6	9,5E+0	5,00E+6	9,5E+0
20ms	12,50E+3	6,1E+0	125,00E+3	3,8E+0	1,25E+6	4,8E+0	2,50E+6	4,8E+0
50ms	5,00E+3	2,4E+0	50,00E+3	1,5E+0	500,00E+3	1,9E+0	1,00E+6	1,9E+0
100ms	2,50E+3	1,2E+0	25,00E+3	762,9E-3	250,00E+3	953,7E-3	500,00E+3	953,7E-3
200ms	1,25E+3	610,4E-3	12,50E+3	381,5E-3	125,00E+3	476,8E-3	250,00E+3	476,8E-3
500ms	500,00E+0	244,1E-3	5,00E+3	152,6E-3	50,00E+3	190,7E-3	100,00E+3	190,7E-3
1s	250,00E+0	122,1E-3	2,50E+3	76,3E-3	25,00E+3	95,4E-3	50,00E+3	95,4E-3

SDS1104X-E_2CH_FFT_BW_Step

Why did I limit the timebase range to 10ns – 1s? The scope can certainly do a much wider range? The lower limit is 10ns/div just because this already means a FFT length of only 64 points in dual channel mode. This is about the limit for any useful FFT and there is certainly no advantage whatsoever going any lower. It is different for the upper limit and there is basically nothing wrong with timebase settings slower than 1s/div. This would allow us to get extremely narrow frequency steps and resolution bandwidths in turn. But then, even with just 1s/div, one single acquisition already takes a healthy 14 seconds and frequency steps down to some 0.1Hz are obtained. I felt this should cover the vast majority of use cases.

If only one channel in a group is enabled, the max. sample rate as well as memory depth will be doubled. During my experiments, I got the impression that this operating mode generates slightly less spurious signals. The table for single channel (interleaved) mode is shown below.



SDS1104X-E Single Channel FFT Bandwidth / FFT Frequency Step								
	14k		140k		1.4M		14M	
Timebase	BW [Hz]	df [Hz]	BW [Hz]	df [Hz]	BW [Hz]	df [Hz]	BW [Hz]	df [Hz]
10ns	500,00E+6	7,8E+6	500,00E+6	7,8E+6	500,00E+6	7,8E+6	500,00E+6	7,8E+6
20ns	500,00E+6	3,9E+6	500,00E+6	3,9E+6	500,00E+6	3,9E+6	500,00E+6	3,9E+6
50ns	500,00E+6	2,0E+6	500,00E+6	2,0E+6	500,00E+6	2,0E+6	500,00E+6	2,0E+6
100ns	500,00E+6	976,6E+3	500,00E+6	976,6E+3	500,00E+6	976,6E+3	500,00E+6	976,6E+3
200ns	500,00E+6	488,3E+3	500,00E+6	488,3E+3	500,00E+6	488,3E+3	500,00E+6	488,3E+3
500ns	500,00E+6	244,1E+3	500,00E+6	244,1E+3	500,00E+6	244,1E+3	500,00E+6	244,1E+3
1µs	500,00E+6	122,1E+3	500,00E+6	122,1E+3	500,00E+6	122,1E+3	500,00E+6	122,1E+3
2µs	250,00E+6	61,0E+3	500,00E+6	61,0E+3	500,00E+6	61,0E+3	500,00E+6	61,0E+3
5µs	100,00E+6	24,4E+3	500,00E+6	15,3E+3	500,00E+6	15,3E+3	500,00E+6	15,3E+3
10µs	50,00E+6	12,2E+3	500,00E+6	7,6E+3	500,00E+6	7,6E+3	500,00E+6	7,6E+3
20µs	25,00E+6	6,1E+3	250,00E+6	3,8E+3	500,00E+6	3,8E+3	500,00E+6	3,8E+3
50µs	10,00E+6	2,4E+3	100,00E+6	1,5E+3	500,00E+6	1,9E+3	500,00E+6	1,9E+3
100µs	5,00E+6	1,2E+3	50,00E+6	762,9E+0	500,00E+6	953,7E+0	500,00E+6	953,7E+0
200µs	2,50E+6	610,4E+0	25,00E+6	381,5E+0	250,00E+6	476,8E+0	250,00E+6	476,8E+0
500µs	1,00E+6	244,1E+0	10,00E+6	152,6E+0	100,00E+6	190,7E+0	100,00E+6	190,7E+0
1ms	500,00E+3	122,1E+0	5,00E+6	76,3E+0	50,00E+6	95,4E+0	50,00E+6	95,4E+0
2ms	250,00E+3	61,0E+0	2,50E+6	38,1E+0	25,00E+6	47,7E+0	25,00E+6	47,7E+0
5ms	100,00E+3	24,4E+0	1,00E+6	15,3E+0	10,00E+6	19,1E+0	10,00E+6	19,1E+0
10ms	50,00E+3	12,2E+0	500,00E+3	7,6E+0	5,00E+6	9,5E+0	5,00E+6	9,5E+0
20ms	25,00E+3	6,1E+0	250,00E+3	3,8E+0	2,50E+6	4,8E+0	2,50E+6	4,8E+0
50ms	10,00E+3	2,4E+0	100,00E+3	1,5E+0	1,00E+6	1,9E+0	1,00E+6	1,9E+0
100ms	5,00E+3	1,2E+0	50,00E+3	762,9E-3	500,00E+3	953,7E-3	500,00E+3	953,7E-3
200ms	2,50E+3	610,4E-3	25,00E+3	381,5E-3	250,00E+3	476,8E-3	250,00E+3	476,8E-3
500ms	1,00E+3	244,1E-3	10,00E+3	152,6E-3	100,00E+3	190,7E-3	100,00E+3	190,7E-3
1s	500,00E+0	122,1E-3	5,00E+3	76,3E-3	50,00E+3	95,4E-3	50,00E+3	95,4E-3

SDS1104X-E_1CH_FFT_BW_Step

IMPORTANT: Please note that these tables are not guaranteed to be entirely correct as they just contain calculation results and not collected data from the real scope – it would have been rather time consuming to actually try all these combinations on the instrument. The sample rate in the SDS1kX-E might not always be in accordance with my calculations, yet most results should be correct and the tables good enough for determining the appropriate settings for a certain bandwidth / frequency step combination.

For even greater convenience, I've prepared a set of tables that show all appropriate settings for any available combination of analysis bandwidth & frequency step. To use these tables, proceed as follows:

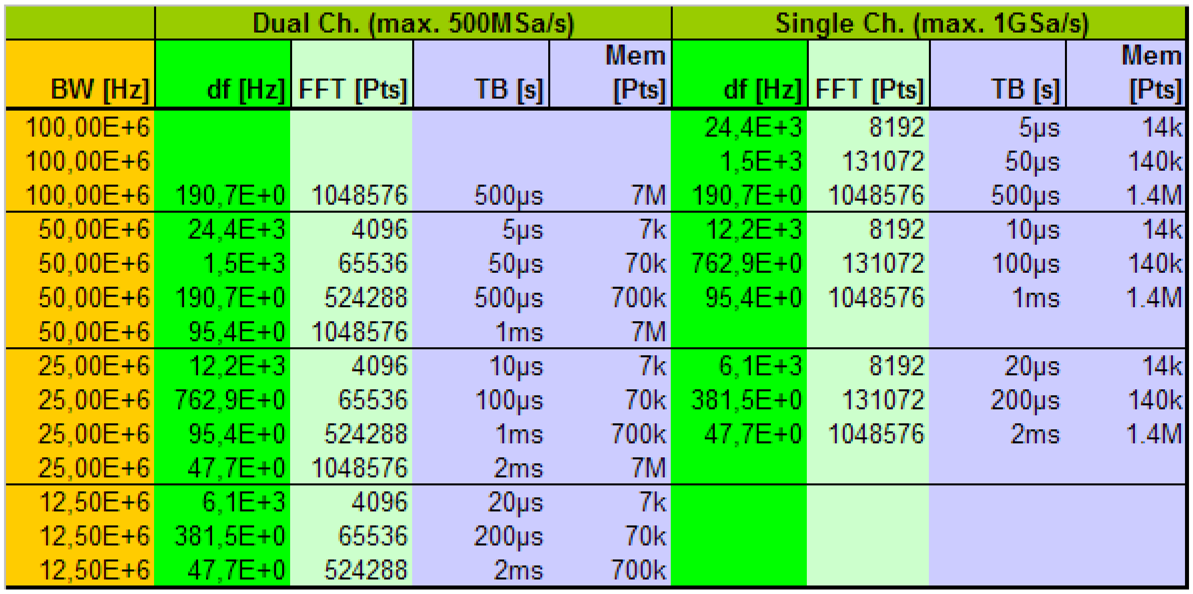
1. Look at all table entries that show the desired analysis bandwidth in column **BW [Hz]**.
2. Pick the entry with the desired frequency step in column **df [Hz]**. Note that there are two such columns, one for dual channel (individual) and another one for single channel (interleaved) configuration.
3. Use the associated entries for timebase **TB [s]** and Max. memory depth **Mem [Pts]** to configure the scope accordingly.
4. The **FFT [Pts]** entry is there just as additional information about the actual FFT length.

	Dual Ch. (max. 500MSa/s)				Single Ch. (max. 1GSa/s)			
BW [Hz]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]
500,00E+6					7,8E+6	128	10ns	≥14k
500,00E+6					3,9E+6	256	20ns	≥14k
500,00E+6					2,0E+6	512	50ns	≥14k
500,00E+6					976,6E+3	1024	100ns	≥14k
500,00E+6					488,3E+3	2048	200ns	≥14k
500,00E+6					244,1E+3	4096	500ns	≥14k
500,00E+6					122,1E+3	8192	1µs	≥14k
500,00E+6					61,0E+3	16384	2µs	≥140k
500,00E+6					15,3E+3	65536	5µs	≥140k
500,00E+6					7,6E+3	131072	10µs	≥140k
500,00E+6					3,8E+3	262144	20µs	≥1.4M
500,00E+6					1,9E+3	524288	50µs	≥1.4M
500,00E+6					953,7E+0	1048576	100µs	≥1.4M

SDS1104X-E_FFT_Setup_500MHz

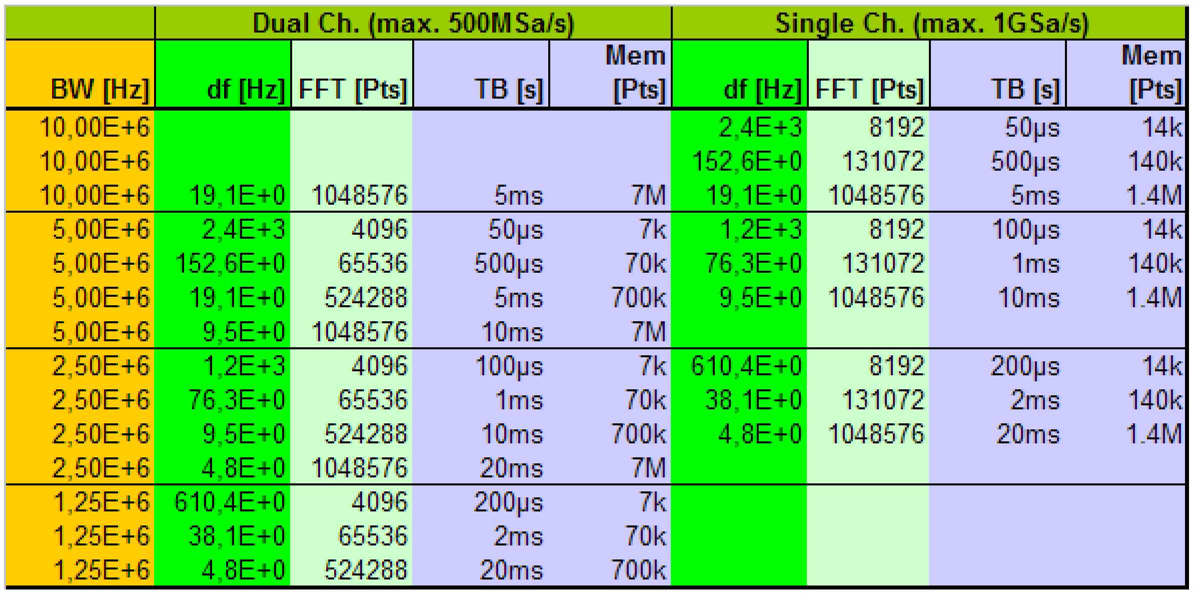
	Dual Ch. (max. 500MSa/s)				Single Ch. (max. 1GSa/s)			
BW [Hz]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]
250,00E+6	7,8E+6	64	10ns	≥7k				
250,00E+6	3,9E+6	128	20ns	≥7k				
250,00E+6	2,0E+6	256	50ns	≥7k				
250,00E+6	976,6E+3	512	100ns	≥7k				
250,00E+6	488,3E+3	1024	200ns	≥7k				
250,00E+6	244,1E+3	2048	500ns	≥7k				
250,00E+6	122,1E+3	4096	1µs	≥7k				
250,00E+6	61,0E+3	8192	2µs	≥70k	61,0E+3	8192	2µs	14k
250,00E+6	15,3E+3	32768	5µs	≥70k				
250,00E+6	7,6E+3	65536	10µs	≥70k				
250,00E+6	3,8E+3	131072	20µs	≥700k	3,8E+3	131072	20µs	140k
250,00E+6	1,9E+3	262144	50µs	≥700k				
250,00E+6	953,7E+0	524288	100µs	≥700k				
250,00E+6	476,8E+0	1048576	200µs	7M	476,8E+0	1048576	200µs	1.4M
125,00E+6	61,0E+3	4096	2µs	7k				
125,00E+6	3,8E+3	65536	20µs	70k				
125,00E+6	476,8E+0	524288	200µs	700k				

SDS1104X-E_FFT_Setup_125-250MHz



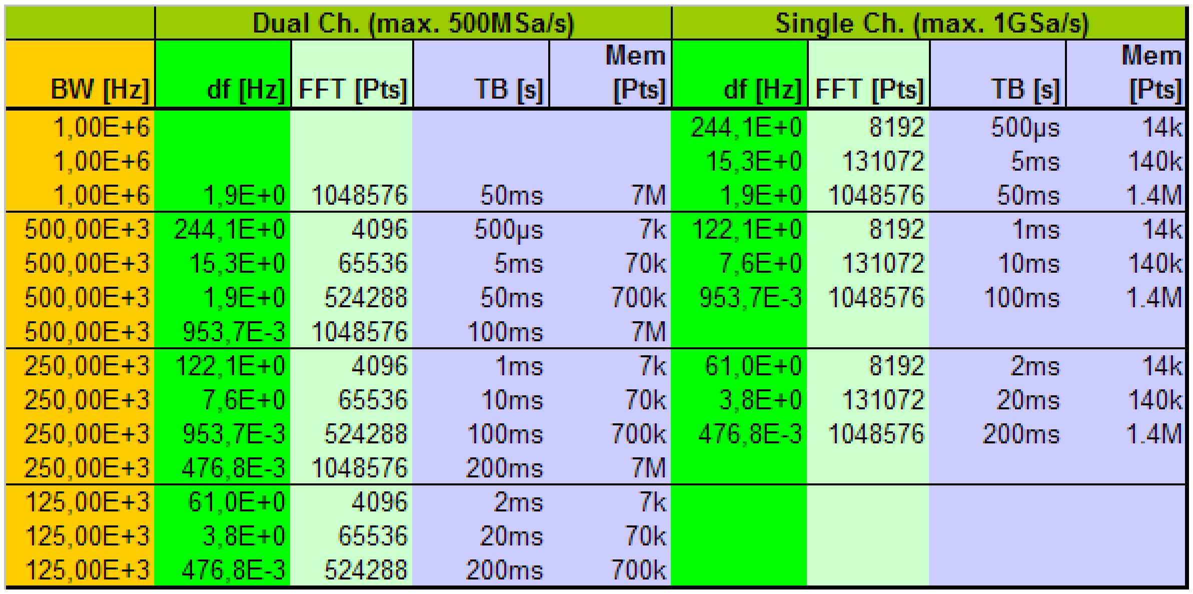
	Dual Ch. (max. 500MSa/s)				Single Ch. (max. 1GSa/s)			
BW [Hz]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]
100,00E+6					24,4E+3	8192	5µs	14k
100,00E+6					1,5E+3	131072	50µs	140k
100,00E+6	190,7E+0	1048576	500µs	7M	190,7E+0	1048576	500µs	1.4M
50,00E+6	24,4E+3	4096	5µs	7k	12,2E+3	8192	10µs	14k
50,00E+6	1,5E+3	65536	50µs	70k	762,9E+0	131072	100µs	140k
50,00E+6	190,7E+0	524288	500µs	700k	95,4E+0	1048576	1ms	1.4M
50,00E+6	95,4E+0	1048576	1ms	7M				
25,00E+6	12,2E+3	4096	10µs	7k	6,1E+3	8192	20µs	14k
25,00E+6	762,9E+0	65536	100µs	70k	381,5E+0	131072	200µs	140k
25,00E+6	95,4E+0	524288	1ms	700k	47,7E+0	1048576	2ms	1.4M
25,00E+6	47,7E+0	1048576	2ms	7M				
12,50E+6	6,1E+3	4096	20µs	7k				
12,50E+6	381,5E+0	65536	200µs	70k				
12,50E+6	47,7E+0	524288	2ms	700k				

SDS1104X-E_FFT_Setup_12.5-100MHz



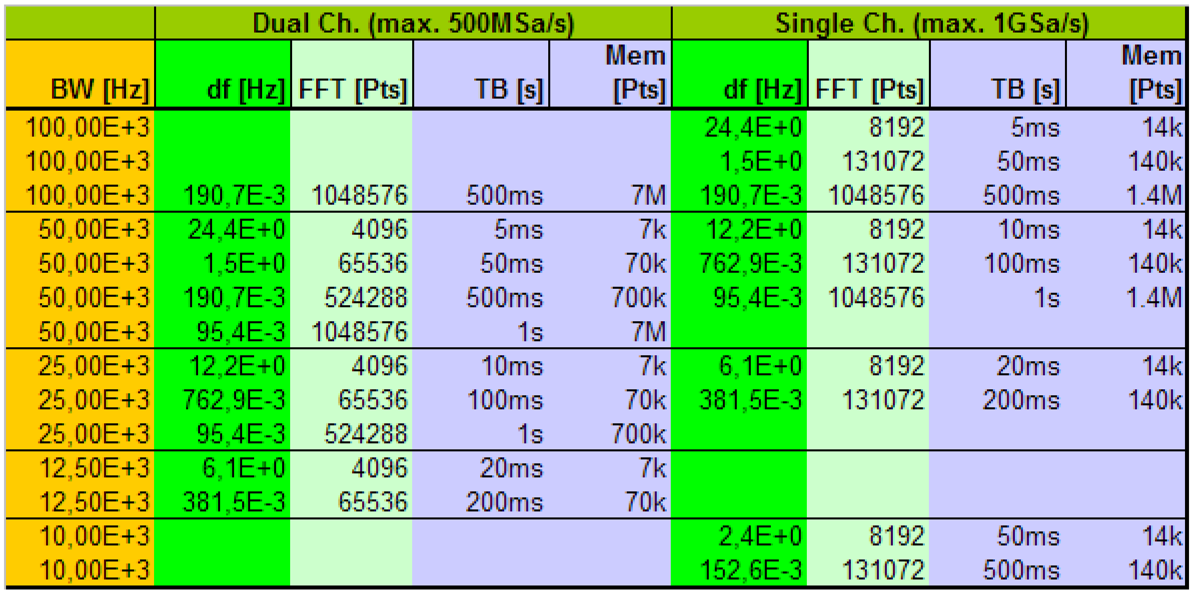
	Dual Ch. (max. 500MSa/s)				Single Ch. (max. 1GSa/s)			
BW [Hz]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]
10,00E+6					2,4E+3	8192	50µs	14k
10,00E+6					152,6E+0	131072	500µs	140k
10,00E+6	19,1E+0	1048576	5ms	7M	19,1E+0	1048576	5ms	1.4M
5,00E+6	2,4E+3	4096	50µs	7k	1,2E+3	8192	100µs	14k
5,00E+6	152,6E+0	65536	500µs	70k	76,3E+0	131072	1ms	140k
5,00E+6	19,1E+0	524288	5ms	700k	9,5E+0	1048576	10ms	1.4M
5,00E+6	9,5E+0	1048576	10ms	7M				
2,50E+6	1,2E+3	4096	100µs	7k	610,4E+0	8192	200µs	14k
2,50E+6	76,3E+0	65536	1ms	70k	38,1E+0	131072	2ms	140k
2,50E+6	9,5E+0	524288	10ms	700k	4,8E+0	1048576	20ms	1.4M
2,50E+6	4,8E+0	1048576	20ms	7M				
1,25E+6	610,4E+0	4096	200µs	7k				
1,25E+6	38,1E+0	65536	2ms	70k				
1,25E+6	4,8E+0	524288	20ms	700k				

SDS1104X-E_FFT_Setup_1.25-10MHz



	Dual Ch. (max. 500MSa/s)				Single Ch. (max. 1GSa/s)			
BW [Hz]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]
1,00E+6					244,1E+0	8192	500µs	14k
1,00E+6					15,3E+0	131072	5ms	140k
1,00E+6	1,9E+0	1048576	50ms	7M	1,9E+0	1048576	50ms	1.4M
500,00E+3	244,1E+0	4096	500µs	7k	122,1E+0	8192	1ms	14k
500,00E+3	15,3E+0	65536	5ms	70k	7,6E+0	131072	10ms	140k
500,00E+3	1,9E+0	524288	50ms	700k	953,7E-3	1048576	100ms	1.4M
500,00E+3	953,7E-3	1048576	100ms	7M				
250,00E+3	122,1E+0	4096	1ms	7k	61,0E+0	8192	2ms	14k
250,00E+3	7,6E+0	65536	10ms	70k	3,8E+0	131072	20ms	140k
250,00E+3	953,7E-3	524288	100ms	700k	476,8E-3	1048576	200ms	1.4M
250,00E+3	476,8E-3	1048576	200ms	7M				
125,00E+3	61,0E+0	4096	2ms	7k				
125,00E+3	3,8E+0	65536	20ms	70k				
125,00E+3	476,8E-3	524288	200ms	700k				

SDS1104X-E_FFT_Setup_125kHz-1MHz



	Dual Ch. (max. 500MSa/s)				Single Ch. (max. 1GSa/s)			
BW [Hz]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]	df [Hz]	FFT [Pts]	TB [s]	Mem [Pts]
100,00E+3					24,4E+0	8192	5ms	14k
100,00E+3					1,5E+0	131072	50ms	140k
100,00E+3	190,7E-3	1048576	500ms	7M	190,7E-3	1048576	500ms	1.4M
50,00E+3	24,4E+0	4096	5ms	7k	12,2E+0	8192	10ms	14k
50,00E+3	1,5E+0	65536	50ms	70k	762,9E-3	131072	100ms	140k
50,00E+3	190,7E-3	524288	500ms	700k	95,4E-3	1048576	1s	1.4M
50,00E+3	95,4E-3	1048576	1s	7M				
25,00E+3	12,2E+0	4096	10ms	7k	6,1E+0	8192	20ms	14k
25,00E+3	762,9E-3	65536	100ms	70k	381,5E-3	131072	200ms	140k
25,00E+3	95,4E-3	524288	1s	700k				
12,50E+3	6,1E+0	4096	20ms	7k				
12,50E+3	381,5E-3	65536	200ms	70k				
10,00E+3					2,4E+0	8192	50ms	14k
10,00E+3					152,6E-3	131072	500ms	140k

SDS1104X-E_FFT_Setup_10-100kHz

As an example, let's assume we want to perform an analysis in the audio range up to 20kHz, which would then be our desired analysis bandwidth. In the last table SDS1104X-E_FFT_Setup_10-100kHz we cannot find 20kHz, so we pick the next higher value, that is 25kHz. For this bandwidth, we get a total of five choices: three for dual channel configuration with frequency steps of 12.2Hz, 0.763Hz and 0.095Hz as well as another two for single channel configuration with frequency steps of 6.1Hz and 0.382Hz. If we use the Blackman window (recommended), we have to multiply the frequency step by 1.7 in order to get the effective -3dB resolution bandwidth. With single channel configuration, 200ms/div and 140k max. memory depth we get a frequency step of 0.3815Hz and the resolution bandwidth will be about 0.65Hz. Acquisition time will be 200ms x 14 = 2.8s, hence FFT update rate will be slow no matter how powerful the signal processing might be. Dealing with low frequencies and narrow frequency steps just takes its toll, there's no way to get around this.

Setting up an FFT Measurement

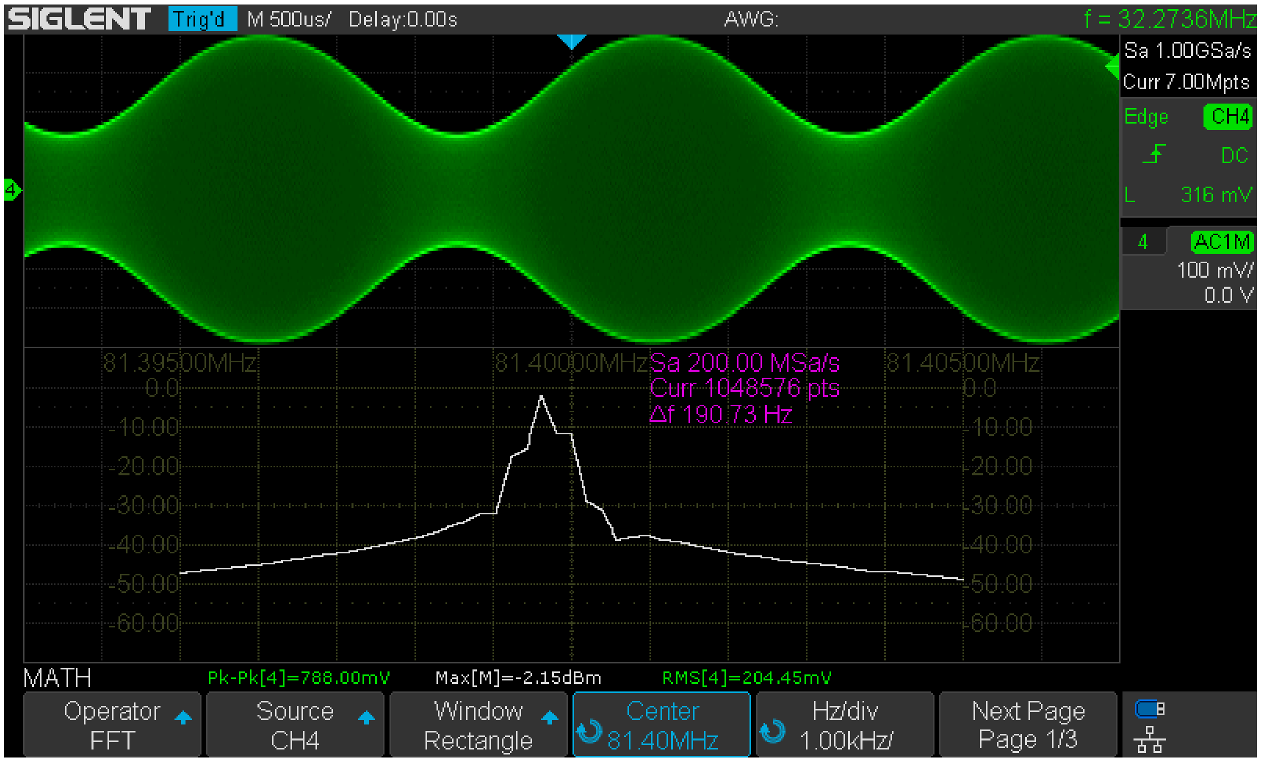
Even from the best FFT implementation, we can only expect good results as long as the scope has been set up properly for that specific task. How many so called "reviews" have we seen where FFT has been engaged and then some scope settings have been randomly altered just to get some halfway plausible but actually rather meaningless FFT graph, which was then either praised or criticized? Of course we can get away with some quick & dirty setup if we just want to get a quick overview, but for optimal speed, frequency resolution and dynamic range, we need to put a little more effort into a proper setup, which has quite different requirements compared to the usual Y-t view. Below there is a complete checklist how to properly set up the DSO for analysis in the frequency domain:

1. Set acquisition mode to normal. Use average only for a good reason and stay away from Eres. Avoid Peak Detect under all circumstances and without any exception!
2. Use edge trigger in auto mode to make sure signal acquisition doesn't stop even when the signal amplitude drops below the trigger sensitivity. FFT doesn't absolutely require a stable trigger by the way, but it certainly doesn't hurt, especially for narrowband analysis.
3. Determine the lower bandwidth limit for the FFT analysis. If it is >3Hz, use AC-coupling for the input channel to ensure maximum dynamic range even with large DC offsets and/or high input sensitivities.
4. Determine the upper bandwidth limit for the FFT analysis. In order to avoid aliasing artifacts, this should not only cover the desired analysis bandwidth, but include the highest expected input frequency. In general, it's best to start with a higher upper bandwidth limit and reduce it only after it has been confirmed that there is no significant signal content above the desired final limit.
5. Choose the frequency step size, which would be about half the required resolution bandwidth.
6. Find an appropriate set of horizontal timebase and max. memory depth settings by means of the tables provided in the *FFT-Bandwidth and RBW* section earlier in this document and setup the scope accordingly. Be aware that the desired resolution bandwidth might not be achievable due to the limited choice of sample rates and memory depths and/or the maximum FFT length of 1Mpts.
7. Engage FFT mode, select the correct source channel and start with Split Screen mode.
8. Set the vertical gain so that the peak amplitude of the input signal is between ±2 to ±4 divisions.
9. Set the FFT center frequency to the arithmetic mean between lower and upper bandwidth limit.
10. Set the FFT frequency scale so that the desired analysis bandwidth is displayed on the screen.
11. Set the desired level units and make sure the external load impedance matches reality whenever working with power levels, i.e. dBm.
12. Set the reference level and vertical scale so that the FFT amplitude range of interest makes best use of the available space.
13. Setup automatic peak-peak (and maybe RMS) measurement for the input channel, as well as Max for the math channel. During frequency domain analysis, especially in Exclusive mode, keep an eye on the Vpp measurement for the input channel to make sure no overload occurs.
14. Select an appropriate window function; refer to the *Window* section earlier in this document.

Hint: stay in Split Screen mode until the amplitude setup is finished and the levels are reasonably stable, then switch to Exclusive mode. By keeping an eye on the peak to peak measurement of the input signal, you can still detect an overload condition instantly; the scope indicates that by displaying **>** instead of **=** in front of the measurement value, e.g. Pk-Pk[4]**>**796.00mV instead of Pk-Pk[4]**=**640.00mV.

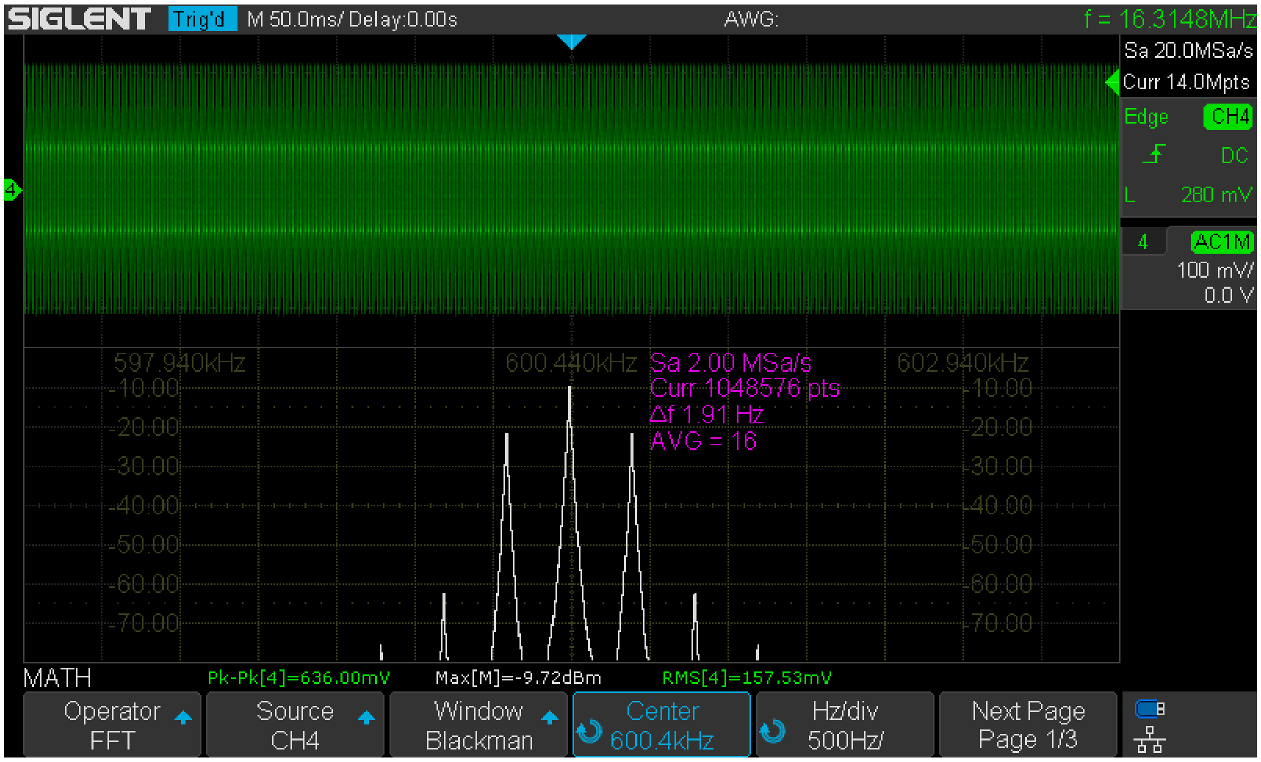
Looking at IF Signals

Sometimes we want more resolution bandwidth than a 1Mpts FFT can provide. Assume a professional HF communications receiver that has an IF of 81.4MHz, where we want to analyze a 400Hz AM signal within the IF signal chain. We could try to do that by setting up a 100MHz analysis bandwidth and using maximum FFT length. According to table SDS1104X-E_FFT_Setup_12.5-100MHz this can be done by setting the timebase to 500µs/div and max. memory depth to (at least) 1.4Mpts in single channel (interleaved) mode. This results in a frequency step of 190.7Hz, which is just too wide for properly analyzing sidebands that are only 400Hz away from the carrier.



SDS1104X-E_FFT_AM_400Hz_81.4MHz

Even with the rectangle window (which gives the most narrow resolution bandwidth), we don’t get enough frequency resolution to distinguish the sidebands from the carrier. We can try something different though:



SDS1104X-E_FFT_AM_400Hz_81.4MHz_US

Since an IF signal is bandwidth limited (hence no risk of aliasing), we can downconvert it just by undersampling. Any ADC acts as a mixer, thus producing a spectrum of ±*n* x *fi* ± *m* x *fs*, where *fi* is the input frequency and *fs* is the sample clock, whereas *n* and *m* are just integers running from 0 to (theoretically) infinity. During normal operation, we don't want to see any mixer products, which is perfectly possible as long as the input signal and all its harmonics don't exceed *fs*/2 and the output of the ADC has a brick-wall filter (then in the digital domain of course) that removes everything above *fs*/2. But in some circumstances, we can make use of a certain high-order mixer product, just as in this example, where the effective FFT sample rate is only 2MSa/s, which is obviously much too low for an 81.4MHz signal.

According to the formula given above, we are aiming at the mixer product for -1 x *fi* + 41 x *fs*, which is

41 x 2MHz – 1 x 81.4MHz = 82MHz – 81.4MHz = 600kHz;

Now if we set the center frequency to 600kHz, we get the carrier at 81.4MHz and can also clearly see the sidebands 400Hz apart from it. 16x Averaging has been used in order to get a clean and stable display.

There are several drawbacks though:

- With this mixing scheme, we get the result in reverse frequency position, i.e. the upper sideband appears below the carrier and vice versa.
- Mixing with the 41st harmonic of the sample clock introduces also 41 times more phase noise and jitter and it clearly shows in the FFT plot. Just look at the filter shape of the individual spectral lines.
- Amplitude accuracy is pretty much gone, as a harmonic mixing process cannot be as efficient as the fundamental one, hence we get about 4.6dB attenuation. Note that I had to reduce the input level by 3dB compared to the first screenshot in order to avoid input overloading, so 3dB of the total difference are caused by that and not by the measurement error due to harmonic mixing.

So this is not an ideal application, just some emergency measure to get a result – following the motto "a compromised measurement is better than nothing at all …"

Performance Test

Up to now, we have just explored the possibilities to properly set up an optimal FFT analysis for various tasks, but we have not checked the actual performance of the FFT implementation in the SDS1104X-E. We can often hear opinions suggesting that the FFT length is the most important factor and if this were true, the Siglent SDS1kX-E series with 1Mpts max. FFT length would be hard to beat. Actually, FFT length only determines the frequency resolution and noise. Yet for the majority of tasks, like characterizing an unknown signal and interference hunting, we don't really need an extremely high frequency resolution. Likewise, for the vast majority of measurements with a DSO, noise isn't the limiting factor either. Consequently, a long FFT is nice to have, but most of the time a high spurious-free dynamic range and low harmonic and intermodulation distortions would be much more important. For this, an 8-bit sampling system sets tight limits with about 49dB dynamic range, which cannot be changed by increasing the FFT length or any other averaging techniques. We can indeed get more than that in certain scenarios and an e.g. 70dB dynamic could easily be demonstrated with a special setup, but this is not universally true and ironically fails just for the majority of real world applications. So while we should not expect any wonders, the FFT in this scope is still a very powerful implementation and very useful tool within the constraints of the 8-bit acquisition system.

Amplitude Accuracy

Analyzing a signal in the frequency domain is about exploring its spectral components with their amplitudes and maybe also relative phases. Well, we obviously don't get any phase information and this is quite common in DSO-FFT implementations as well as scalar spectrum analyzers. But we do expect decent amplitude accuracy within the dynamic range of the 8-bit system, at least when using the Flattop window. For this test, a 50MHz sine from a DDS function generator is fed into channel 4 of the scope via a precision step attenuator and this combination is capable of providing a fairly accurate amplitude range of more than 120dB. Let's start with 0dBm using the low noise gain of 100mV/div and an analysis bandwidth of 100MHz so we can see all the spurious signals generated by the scope itself.