# PicoHarp 330

High Resolution Event Timer

PicoQuant

Multichannel Time–Correlated Single Photon Counting and Time Tagging System with USB Interface

User's Manual and Technical Data

Version 2.0.0.0

Disclaimer

PicoQuant GmbH disclaims all warranties with regard to the supplied software and documentation including all implied warranties of merchantability and fitness for a particular purpose. In no case shall PicoQuant GmbH be liable for any direct, indirect or consequential damages or any material or immaterial damages whatsoever resulting from loss of data, time or profits; arising from use, inability to use, or performance of this software and associated documentation.

License and Copyright Notice

With the PicoHarp 330 hardware product you have purchased a license to use the PicoHarp 330 software. You have not purchased any other rights to the software itself. The software is protected by copyright and intellectual property laws. You may not distribute the software to third parties or reverse engineer, decompile or disassemble the software or part thereof. You may use and modify demo code to create your own software. Original or modified demo code may be re–distributed, provided that the original disclaimer and copyright notes are not removed from it. Copyright of this manual and on–line documentation belongs to PicoQuant GmbH. No parts of it may be reproduced, translated or transferred to third parties without written permission of PicoQuant GmbH.

Products and corporate names appearing in this manual may or may not be registered trademarks or subject to copyrights of their respective owners. PicoQuant GmbH claims no rights to any such trademarks. They are used here only for identification or explanation and to the owner's benefit, without intent to infringe.

Acknowledgments

When the PicoHarp 330 software is used under Linux it uses Libusb to access the PicoHarp 330 USB devices. Libusb is licensed under the LGPL which allows a fairly free use even in commercial projects. For details and precise terms please see http://libusb.info. In order to meet the license requirements a copy of the LGPL as applicable to Libusb is provided as part of the PicoHarp 330 software distribution package. The LGPL does not apply to the PicoHarp 330 software as a whole.

Table of Contents

1. Introduction

While intensity based fluorescence spectroscopic investigations have been common for a long time, extracting additional temporal information from quantum systems via pulsed excitation and time-resolved detection is a more powerful technique. The temporal analysis can reveal information about the emitter that is not available from spectral data alone. This is why time-resolved analysis of (typically laser induced) fluorescence by means of Time–Correlated Single Photon Counting (TCSPC) has gained in importance over the recent years. For instance, in life sciences the difference in the fluorescence decay times of fluorophores provides a powerful discrimination feature to distinguish molecules of interest from background or other species. This has made the technique very interesting for sensitive analysis, even down to the single molecule level. The same mechanisms are applicable in quantum optics, e.g., when quantum dots or defect centers in diamond are observed.

The acquisition of fluorescence decay curves by means of TCSPC provides temporal resolution and sensitivity that cannot easily be achieved with other methods. In practice this is done by histogramming arrival times of individual photons over many excitation and fluorescence cycles. The arrival times recorded in the histogram are relative times between excitation and corresponding fluorescence photon arrival (start / stop times) which can be resolved down to a few picoseconds. The resulting histogram represents the fluorescence decay. Although fluorescence lifetime analysis is a great field of application for the PicoHarp 330, it is in no way restricted to this task. Other important applications are e.g. Quantum Optics, Quantum Cryptography (QC) Time–Of–Flight (TOF) and Optical Time Domain Reflectometry (OTDR) as well as any kind of coincidence correlation.

The PicoHarp 330 is a cutting edge TCSPC and time tagging system with USB interface. Its new integrated design provides a flexible number of input channels at very reasonable cost and enables innovative measurement approaches. The timing circuits allow high measurement rates up to 80 million counts per second (Mcps) each, with an excellent time resolution of 1 ps, extremely low timing Jitter and a record breaking dead time of 680 ps. The latest Model PicoHarp 330 4P also features a very high precision internal clock source with a frequency accuracy of ±300 ppb and a frequency stability of ±10 ppb. The USB 3.0[1] super speed interface provides very high throughput as well as 'plug and play' installation. The input circuits can be programmed to act as edge triggers or as Constant Fraction Discriminators (CFD) and are adjustable for a wide range of input signals. These specifications qualify the PicoHarp 330 for use with a large variety of phtoton detectos, such as Superconducting Nanowire Single Photon Detectors (SNSPD), Single Photon Avalanche Diodes (SPADs), Hybrid Photodetectors (HPD), and Photomultiplier Tubes (PMT). Depending on detector and excitation source the FWHM of the overall Instrument Response Function (IRF) can be as small as 15 ps. The PicoHarp 330 can be purchased with up to four timing inputs and one synchronization (sync) input. The use of these inputs is very flexible. In fluorescence lifetime applications the sync channel is typically used as a synchronization input from a laser. The other inputs are then used for photon detectors. Alternatively, notably in quantum optics applications, all inputs including the sync input can be used for photon detectors.

The PicoHarp 330 can operate in various modes to adapt to different measurement needs. Two different Time–Tagged–Time–Resolved (TTTR) modes allow recording of each photon event on separate, independent channels, thereby providing unlimited flexibility in off–line data analysis such as burst detection and time–gated or lifetime weighted Fluorescence Correlation Spectroscopy (FCS) as well as picosecond coincidence correlation, using the individual photon arrival times. The standard histogram mode performs real–time histogramming in software. The PicoHarp 330 is supported by a variety of accessories such as pre–amplifiers, signal adapters and detector assemblies from PicoQuant.

The PicoHarp 330 software runs on current x64 Windows PC platforms and even on Linux with Wine. It provides functions for setting measurement parameters, performing measurements, displaying measurement results as well as loading and saving of measurement parameters and results. Important measurement characteristics such as count rate, count maximum and position as well as histogram width (FWHM) are displayed and updated continuously. Data can conveniently be exported via the clipboard, e.g. for immediate processing by the PicoQuant EasyTau 2 fluorescence decay fit software. Furthermore, a programming library (DLL) enables users to write custom data acquisition programs for the PicoHarp 330 in virtually all popular programming environments. There is also a library version for Linux (Intel processor architecture only) which is fully compatible with that for Windows so that applications can easily be ported across the two platforms.

For details on the Time–Correlated Single Photon Counting method, please read the next section as well as our TechNote on TCSPC and consult the literature referenced at the end of section 2.4. Experienced users of the method should be able to work with the PicoHarp 330 straight away. Nevertheless, we recommend carefully reading sections 3.4 and 3.5 on hardware and software installation to avoid damage. Later, the comprehensive online–help function of the PicoHarp 330 software will probably let the manual gather dust on the shelf.

---

1   The latest USB specifications have introduced new naming schemes where the original USB 3.0 (5 GBits/s) was later called "USB 3.1 Gen 1" and now "USB 3.2. Gen 1x1". We stick to the original name USB 3.0 throughout this document.

2.　Primer on Time–Correlated Single Photon Counting

In order to make use of a powerful analysis tool such as time–resolved fluorescence spectroscopy, the time dependent intensity of the emitted light must be recorded. While in principle it could be attempted to record to record the intensity decay of the signal from a single excitation / emission cycle, there are practical problems that prevent such a simple solution in most cases. First of all, the decay to be recorded is usually very fast. Typical fluorescence from organic fluorophores may last only a few tens of picoseconds to some hundred nanoseconds. In order to recover fluorescence lifetimes as short as e.g., 100 ps, it is necessary to resolve the recorded signal at least to such an extent, that the exponential decay is represented by enough sample points in time. This means that the required transient recorder would have to sample at very high rates. This is hard to achieve with ordinary electronic transient recorders of reasonable dynamic range. Secondly, the light available may simply be too weak to sample an analog intensity decay. Indeed the signal may consist of just single photons per excitation / emission. This is typically the case for single molecule experiments or work with very small sample volumes / concentrations. Then the discrete nature of the signal itself prohibits analog sampling. Even if more than just a single molecule is measured and the excitation power is increased to obtain more fluorescence light, there are other limits, e.g. due to losses in collection optics, spectral limits of detector sensitivity or photo–bleaching at higher excitation power. The solution is Time–Correlated Single Photon Counting (TCSPC). By using periodic excitation (typically from a laser) it is possible to extend the data collection over multiple excitation/ emission cycles upon which the single cycle decay profile can be reconstructed from single photon events collected over many cycles.

The TCSPC method is based on the repetitive, precisely timed registration of single photons of e.g., a fluorescence signal. The reference for the timing is the corresponding excitation pulse. A single photon detector such as a Photo Multiplier Tube (PMT) or a Single Photon Avalanche Photodiode (SPAD) is used to capture the fluorescence photons. Provided that the probability of registering more than one photon at a time is low, the classic TCSPC histogram of photon arrivals per time bin represents the time decay that would have been obtained from a single shot time–resolved analog recording. The precondition of single photon probability can be met by attenuating the light level reaching the sample. If the single photon probability condition is met by means of atteuation, there may actually be no photons registered in many of the excitation cycles. The diagrams below illustrate how the histogram is formed over multiple cycles.
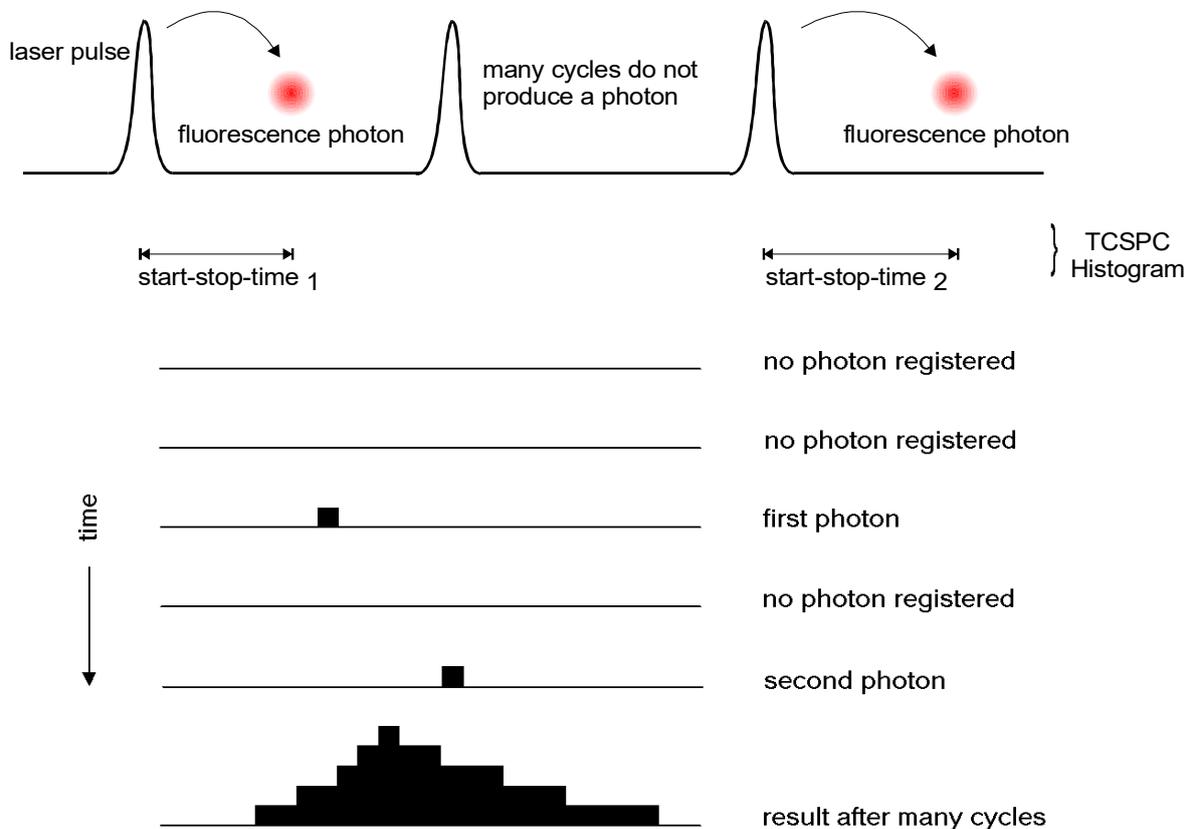
The histogram is collected in a block of memory, where one memory cell holds the photon counts for one corresponding time bin. These time bins are often (historically) referred to as "time channels". In practice, the registration of one photon involves the following steps: first, the time difference between the photon event and the corresponding excitation pulse must be measured. For this purpose both optical signals are converted to electrical signals. For the excitation pulse this can be done via another detector. Alternatively, an electrical sync signal can be used, which is usally supplied directly by most modern lasers. Obviously, all conversion to electrical pulses must preserve the precise timing of the signals as accurately as possible. The actual time difference measurement is done by means of fast electronics which provide a digital timing result. This digital timing result is then used to address the histogram memory so that each possible timing value corresponds to one memory cell or histogram bin. Finally the addressed histogram cell is incremented. All steps are carried out by fast electronics or software so that the processing time required for each photon event is as short as possible. When sufficient counts have been collected, the histogram memory can be read out. The histogram data can then be used for display and e.g., fluorescence lifetime calculation. In the following we will expand on the various steps involved in the method and related issues of importance.

2.1.   Count Rates and Single Photon Statistics

It was mentioned that in the past it was necessary to maintain a low probability of registering more than one photon per cycle. This condition ensured that the histogram of photon arrivals represents the time decay that is obtained from a single shot time–resolved analog recording. This was necessary, as historical TCSPC systems could register only one photon per excitation / emission cycle, due to dead times of detector and electronics lasting at least some tens of nanoseconds after a photon event. If then the number of photons occurring in one excitation cycle were typically > 1, the system would very often register the first photon but miss the following ones. This would lead to an over-representation of early photons in the histogram, an effect called 'pile-up'. This would distortthe recording of the fluorescence decay, typically making the fluorescence lifetime appear shorter. It was therefore crucial to keep the probability of cycles with more than one detected photon sufficiently low, unless one tolerates some error and/or corrects for it in data analysis.

To quantify the "safe" count rate limit for traditional TCSPC, acceptable error limits had to be set and mathematical statistics be applied. For practical purposes a rule of thumb was used: In order to maintain single photon statistics, on average only one in 20 to 100 excitation pulses should generate a count at the detector. In other words: the average count rate at the detector should be at most 1 % to 5 % of the excitation rate. Using e.g., a diode laser from PicoQuant's PDL Series, pulsed at 80 MHz repetition rate, the average detector count rate would then not exceed 4 Mcps. However, high count rates are desired in order to acquire fluorescence decay histograms quickly. This is of particular importance where dynamic lifetime changes or fast molecule transitions are studied or where large numbers of lifetime samples must be collected (e.g., in image scanning). This is why high laser rates (such as 40 or 80 MHz for the PDL Series) are important. Modern detectors can safely handle TCSPC count rates of some tens of Mcps. While old TCSPC electronics would then be limiting the speed of acquisition, newer integrated TCSPC electronics such as the PicoHarp 330 support multi-stop (i.e. recording more than one photon per excitation cycle) and can reach count rates up to 80 Mcps. With the PicoHarp 330 and modern detectors such as Hybrid Photodetectors (HPD) count rates at this upper limit can be collected.

It is worth noting that the photon arrival times are typically random so that there can be bursts of high count rate and periods of low count rates. Bursts may exceed the average rate. The average (sustained) rate the instrument can deal with is important when losses cannot be tolerated, notably in time tagging applications. Due to the high bandwidth of USB 3.0 the PicoHarp 330 can handle sustained time tagging rates as high as 85 Mcps. When comparing count rates considered here and elsewhere please pay attention to the details. The specifications for TCSPC systems often interpret their maximum count rates differently. This is why in this context dead time is also of interest. It describes the time the system cannot register photons while it is processing a previous photon event. The term is applicable both to detectors and electronics. The PicoHarp 330 has an extremely short dead time of about 680 ps, imposing one of the smallest losses among comparable instruments today, only surpassed by PicoQuant's MulitHarp family (650 ps). The short dead time of the PicoHarp 330 together with its multi-stop capability allows measurement scenarios where the classic pile-up limit is no longer critical. This is of special interest for very fast FLIM measurements and PicoQuant's concept of rapidFLIM exploits the idea. Using a HPD (e.g., PicoQuant PMA Hybrid) for detection permits count rates as high as 80 Mcps, enabling the acquisition of high resolution FLIM images at unprecedented speed. While classic pile-up due to dead time is no longer an issue, the effect of pulse-pile-up still has to be considered. This effect occurs when photons arrive at the detector with very short temporal spacing, such that the detector can no longer produce separate output pulses. This merging of detector pulses leads to another type of distortion in the decay histograms. However, a mathematical correction during data analysis ensures that lifetimes and amplitudes are still obtained correctly for very fast quantitative measurements[2]. See section 2.5 or www.picoquant.com for related publications.

---

2   Patent EP3431967

2.2. Timing Resolution

The most critical component for the timing resolution in TCSPC measurements is usually the detector. However, in contrast to analog transient recording, the time resolution of TCSPC is not limited by the impulse response of the detector. Only the timing accuracy of registering a photon determines the resolution. This is limited by the timing uncertainty that the detector introduces in the conversion from a photon to an electrical pulse. This timing error or uncertainty can be as much as ten times smaller than the detector's pulse response. The timing uncertainties are usually quantified by the rms error (standard deviation) or the Full Width at Half Maximum (FWHM) of the timing distribution or instrument response function (IRF). Note that these two notations are related but not identical[3]. Micro channel plate PMTs, can achieve timing uncertainties as small as 25 ps FWHM, while lower cost PMTs or SPADs may introduce uncertainties of 50 to 500 ps FWHM. HPDs lie in between with typical uncertainties of 50..150 ps FWHM. Superconducting nanowire detectors have timing uncertainties of typically 20 to 100 ps FWHM, some optimized designs can even reach below 10 ps.

The second most critical contributor to IRF broadening in fluorescence lifetime measurements with TCSPC is usually the excitation source. While many lasers can provide sufficiently short pulses, it is also necessary to obtain a precise electrical timing reference signal (sync) for comparison with the fluorescence photon signal. The type of sync signal that is available depends on the excitation source. With gain switched diode lasers (e.g., PDL 800–D) a low jitter electrical sync signal is readily available. The sync signal used here is typically a narrow negative pulse of −800 mV into 50 Ω (NIM standard). The sharp falling edge is synchronous with the laser pulse (< 3 ps rms jitter for the PDL 800–D). With other lasers (e.g., Ti:Sa) a second detector must be used to derive a sync signal from the optical pulse train. This is commonly done with a fast photo diode (APD or PIN diode). The light for this reference detector must be derived from the excitation laser beam e.g., by means of a semi–transparent mirror. The reference detector must be chosen and set up carefully as it contributes to the overall timing error.

Another source of timing error is the timing jitter of the electronic components used for TCSPC. This is caused by the finite rise / fall–time of the electrical signals used for the time measurement. At the trigger point of comparators, logic gates etc., the amplitude noise (thermal noise, interference etc.) is always present in these signals and is transformed to a corresponding timing error (phase noise). However, the contribution of the electronics to the total timing error is usually small. For the high resolution device PicoHarp 330 the random jitter of a time difference measurement is typically only 3 ps rms.

Generally, it is desired to keep the pick-up of electrical noise low in all system components. Uncorrelated electrical noise will cause just random jitter and IRF broadening but correlated noise can cause even more detrimental artefacts. This is why signal cables should be properly shielded coax cables, and strong sources of electromagnetic interference should be kept away from the TCSPC electronics and detectors.

The contribution of the time spread introduced by the individual components of a TCSPC system to the total IRF strongly depends on their relative magnitude. Strictly speaking, the overall IRF is the convolution of all component IRFs. An estimate of the overall IRF width, assuming independent noise sources, can be obtained from the geometric sum of the individual components as an rms figure according to statistical error propagation laws:

$$e_{system} = \sqrt{\sum e^2_{component}}$$

As the individual contributions are squared, the total will be dominated by the largest component.

Although it is difficult to specify a general lower limit on the fluorescence lifetime that can be measured by a given TCSPC instrument, as a rule of thumb, one can assume that under favorable conditions lifetimes down to 1 / 10 of the IRF width (FWHM) can still be recovered via deconvolution.

A final time–resolution related issue worth noting here is the bin width of the TCSPC histogram. As outlined above, the analog electronic processing of the timing signals (detector, amplifiers, etc.) creates a continuous distribution around any true time value. In order to form a histogram, at some point the timing results must be quantized. This quantization introduces further error, if chosen too coarse. The quantization step width (i.e. the resolution) should therefore be small compared to the IRF width. As a minimum sampling frequency, from the point of view of information theory, one would consider the Nyquist frequency. That is, the signal should be sampled at least at twice the highest frequency contained in it. For practical purposes one may wish to exceed this limit where possible, but there is usually little benefit in sampling the histogram at resolutions higher than 1 / 10 of the overall IRF width of the analog part of the system.

---

3 In case of a Gaussian distribution the r.m.s deviation corresponds to the standard deviation σ and FWHM ≈ 2.355 σ.

2.3. Photon Counting Detectors

2.3.1. Photomultiplier Tube (PMT)

A PMT consists of a light–sensitive photo cathode that generates electrons when exposed to light. These electrons are directed onto a charged electrode called dynode. The collision of the electrons with the dynode produces additional electrons. Since each electron that strikes the dynode causes several electrons to be emitted, there is a multiplication effect. After further amplification by multiple dynodes, the electrons are collected at the anode of the PMT and output as a current. The current is directly proportional to the light intensity striking the photo cathode. Because of the multiplicative effect of the dynode chain, the PMT is a photo electron amplifier with high sensitivity and remarkably low noise. The high voltage driving the tube may be varied to change the sensitivity of the PMT. Current PMTs have a wide dynamic range, i.e. they can also measure relatively high levels of light. Furthermore, they are very fast, so that rapid successive events can be reliably monitored. One photon on the photo cathode can produce a short output pulse containing millions of photoelectrons. PMTs can therefore be used as single photon detectors. In photon counting mode, individual photons that strike the photo cathode of the PMT are registered. Each photon event gives rise to an electrical pulse at the output. The number of pulses, or counts per second, is proportional to the light impinging upon the PMT. As the number of photon events increase at higher light levels, it will become difficult to differentiate between individual pulses and the photon counting detector's behavior will become non–linear. This usually occurs between 1 and 20 Mcps, depending on the detector design. Similarly, in TCSPC applications, individual photon-generated pulses may merge as the count rate increases. This leads to pulse pile–up and distortions of the collected histograms.

The timing uncertainty between photon arrival and electrical output (transit time spread) is usually small enough to permit time–resolved photon counting at a sub–nanosecond scale. In single photon counting mode the tube is typically operated at a constant high voltage where the PMT is most sensitive.

PMTs usually operate within the blue to red regions of the visible spectrum, with greatest quantum efficiency in the blue–green region, depending upon photo–cathode materials. Typical quantum efficiencies are about 25 %. For spectroscopy experiments in the ultraviolet / visible / near infrared region of the spectrum, a PMT is very well suited.

Because of noise from various sources in the tube, the output of the PMT may contain pulses that are not related to the light input. These are referred to as dark counts. The detection system can to some extent reject these spurious pulses by means of electronic discriminator circuitry. This discrimination is based on the probability that some of the noise generated pulses (those from the dynodes) exhibit lower signal levels than pulses from a true photon event. Thermal emission from the cathode that undergoes the full amplification process can usually not be suppressed this way. In this case cooling of the detector is more helpful.

2.3.2. Micro Channel Plate PMT (MCP)

A Micro Channel Plate PMT consists of an array of glass capillaries (5–25 µm inner diameter) whose insides are coated with an electron–emissive material. The capillaries are biased at a high voltage. Like in a PMT, an electron that strikes the inside wall of one of the capillaries creates an avalanche of secondary electrons. This cascading effect creates a gain of 10^3 to 10^6 and produces a current pulse at the output. Due to the narrow and well defined electron path inside the capillaries, the transit time spread of the output pulses is much reduced compared to a normal PMT. The timing jitter of MCPs is therefore sufficiently small to perform time–resolved photon counting on a picosecond scale, usually outperforming PMTs. Good but also expensive MCPs can achieve timing uncertainties as low as 25 ps. In this respect, microchannel plates are a good match for the PicoHarp 330 but they are quite limited in permitted count rate and provide lower sensitivity towards the red end of the spectrum compared to suitably optimized detectors.

2.3.3. Single Photon Avalanche Photo Diode (SPAD)

Avalanche Photo Diodes (APDs) are photo-sensitive semiconductor devices, usually restricted to the efficient detection of light from the visible to infrared parts of the spectrum. Generally, APDs can be used for ultra–low light detection (optical powers < 1 pW), and may also be used as photon–counters, when operated in the so–called "Geiger" mode (reverse biased slightly above the breakdown voltage). In this case, a single photon is able to trigger an avalanche of about 10^8 charge carriers, enabling the device to be used as a detector for photon counting with very accurate timing of the photon arrival. In this context it can also be referred to as a Single Photon Avalanche Photo Diode (SPAD). Selected devices with small active areas may achieve timing accuracies down to 30 ps, but are usually hard to align and difficult to focus onto. While SPADs are sometimes noisier than PMTs they can also have a greater quantum efficiency especially towards the red side of the spectrum.

Maximum quantum efficiencies are about 70 %. Such sensitive devices provide a timing accuracy of ~400 ps. Commercial modules are thermoelectrically cooled for low dark count rates and deliver pre–shaped TTL pulses. They are the most common detectors for applications where NIR sensitivity is important, e.g., single molecule detection. To achieve the specified timing accuracy, exact focusing onto the center of the active area is necessary. Other SPAD designs such as the PDM family from Micro Photon Devices have the benefit of much better timing resolution and robustness, however, at the expense of a lower sensitivity at the red end of the spectrum.

2.3.4. Hybrid Photodetectors

HPDs make use of a combination of a PMT-like front end followed by an APD structure. Because of the extra gain of the PMT-like front end where a very high voltage accelerates the photoelectrons onto the APD, the latter need not be operated in Geiger mode (reverse bias). While Geiger mode is prone to trapping of charges and their subsequent release in the form of afterpulsing, the operation in linear mode (forward bias) avoids this. The benefits are good timing performance and virtually zero afterpulsing, while the need for very high voltage is a disadvantage when the end user has to take care of it. PicoQuant's PMA Hybrid series include the high voltage supply encapsulated in the detector module as an easy-to-use single package. Another benefit of HPDs is that they do not exhibit a dead time in the strict sense of the word. In principle a HPD can even detect two photons occuring at the same time. However, since the electric output pulse in response to a single photon cannot be made infinitely narrow, there is a limit on the order of 600 ps down to which they can be separated. This results in en effective dead time that is very well matched to that of the PicoHarp 330 so that the method of rapidFLIM can be implemented very efficiently.

2.3.5. Superconducting Nanowire Detectors

Superconducting nanowire single photon detectors (SNSPDs) routinely offer an excellent timing performance (<30 ps jitter) and high sensitivity from the visible to the near infrared with overall system detection efficiencies in excess of 90%. Some timing-optimized designs can even reach below 10 ps but will then typically be less efficient. Current SNSPDs can only operate at cryogenic temperatures, typically between 0.8 K and 4 K, which incurs cost, large footprint and power consumption and makes them less practical for applications where these parameters are of concern. However, more and more compact designs and more detection channels are emerging, so that the limitations become less of an issue. Particularly in quantum optics and applications where sensitivity in the near infrared is important SNSPDs become increasingly popular.

2.3.6. Other Photon Detectors

The field of photon detectors is still evolving. Recent developments apart from improved HPDs and SNSPDs include improved silicon PMTs and new APDs with sufficient gain for single photon detection in analog mode. Each of these detectors have their specific benefits and shortcomings. Only a very brief overview can be given here.

Silicon PMTs are essentially arrays of SPADs, all coupled to a common output. This has the benefit of creating a large area detector that can even resolve photon numbers. The drawback is an increased dark count rate and reduced timing accuracy. Their spectral sensitivity is typically that of a silicon CMOS SPAD as dictated by the CMOS structure depths.

Another class of potentially interesting detectors which have recently emerged are APDs with very high gain. In combination with an electronic amplifier they have been shown to be capable of detecting single photons. As opposed to operation in Geiger mode, this avoids afterpulsing and allows for very high count rates. The disadvantage is a high dark count rate, currently too high for any practical TCSPC application.

2.4. Principles Behind the TCSPC Electronics

For introductory purposes it is worth to look first at the design of historical TCSPC systems. They consist of the following building blocks:

The Constant Fraction Discriminator (CFD) is used to extract precise timing information from the electrical detector pulses that may vary in amplitude. This way the overall system IRF may be tuned to become narrower and some of the random background signal can be suppressed. The same could not be achieved with a simple level trigger (comparator). Especially with PMTs, constant fraction discrimination is important, because their pulse amplitudes vary significantly. In addition, pulses originating from random electrons generated thermally at the dynodes of the PMT can be suppressed. This is because the avalanches evoked by them passed fewer ampliffing steps and their corresponding output pulses are small.

The principle of a classic CFD is the comparison of the original detector signal with an amplified, inverted and delayed version of itself. The signal derived from this comparison changes its polarity exactly when a constant fraction of the detector pulse height is reached. The zero crossing point of this signal is therefore suitable to derive a timing signal independent from the amplitude of the input pulse. In practice the comparison is done by a summation. The timing is done by a subsequent threshold trigger of the sum signal using a settable level, the so called zero cross trigger. Newer CFD designs achieve the same objective by differentiating the input signal and triggering on the zero crossing of the differentiated signal. This has the benefit of adapting to different detector types without a need for changing physical delay lines.

Compared to this, the PicoHarp 330 is different as it allows the configuration of all inputs individually to either use a CFD or a comparator (edge trigger). The user is given a choice and both methods have their advantages and disadvantages. While the comparator is usually faster, its constant trigger level leads to an increased jitter when pulse amplitudes vary significantly, particularly with PMTs and MCPs. This is not the case when using the CFD. However, because of its delay element a CFD requires time to make its "decision". This increases the TCSPC instrument's dead time, during which it cannot process another event. Since a short dead time is a precious feature when using high speed detectors, the use of a CFD would spoil the benefit. Indeed many modern detectors, notably SNSPDs, have very steep signal edges that do not require a CFD. A simple programmable comparator is actually beneficial here. Just like the detector signal, the sync signal must be made available to the timing circuitry. Since the sync pulses are usually of well defined amplitude and shape, an edge trigger is sufficient to accommodate most sync sources. In order to allow the best match for the given setup the PicoHarp 330 permits both configurations.

In historical TCSPC systems the signals from the two input discriminators / triggers are fed to a Time to Amplitude Converter (TAC). This circuit is essentially a highly linear ramp generator that is started by one signal and stopped by the other. The result is a voltage proportional to the time difference between the two signals. In such conventional systems the voltage obtained from the TAC is then fed to an Analog to Digital Converter (ADC) which provides the digital timing value used to address the histogrammer. The ADC must be very fast in order to keep the dead time of the system short. Furthermore it must guarantee a very good linearity (both over the full range as well as differentially). These are criteria difficult to meet simultaneously, particularly with ADCs of high resolution (e.g. 12 bits) as is desirable for TCSPC with many time bins. This is why TACs are rarely used today.
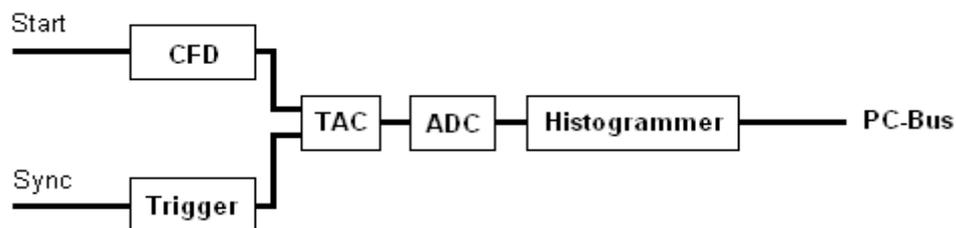
The histogrammer has to increment each histogram memory cell, whose digital address in the histogram memory it receives from the ADC. This is commonly done by fast digital logic e.g., in the form of Field Programmable Gate Arrays (FPGA) or a microprocessor.

While this section so far outlined the typical structure of conventional TCSPC systems, it is important to note that the design of the PicoHarp 330 and all other PicoQuant TCSPC devices is quite different. Today, it is state–of–the–art that the tasks conventionally performed by TAC and ADC are carried out by a so called Time to Digital Converter (TDC). These circuits allow not only timing with picosecond precision but can also extend the measurable time span to virtually any length by means of digital counters. The PicoHarp 330 uses one such circuit in each input channel and one for the sync input. They independently work on each input signal and provide picosecond arrival times that can then be processed further, with a lot more options than in conventional TCSPC systems. In the case of classic TCSPC, this processing consists of a subtraction of the two time figures and histogramming of the differences. This is identical to the classic start–stop measurements of the conventional TAC approach. The following figure exemplifies this for one detector channel.
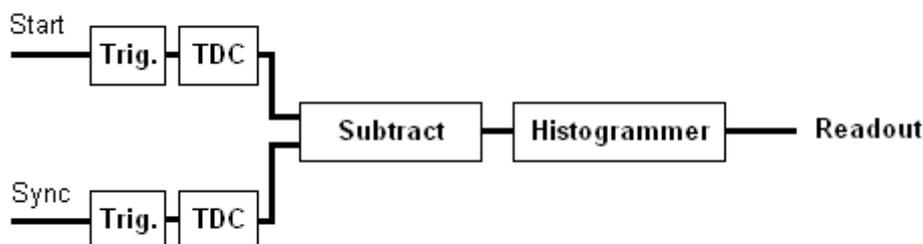
The full strength of the PicoHarp 330 design (and all other PicoQuant TCSPC devices) is exploited by collecting the unprocessed independent arrival times (time tags) as a continuous data stream for more advanced analysis. Details on such advanced analysis can be found in the literature (see section 2.5). For this kind of streaming the device has a large data buffer (First In, First Out; FIFO) so that count rate bursts and irregular data transfer are decoupled. This permits uninterrupted continuous data collection with high throughput. This mode of operation is called Time–Tagged Time–Resolved (TTTR) mode or just "time tagging". Details can be found in section 5.3.

Forward and Reverse Start–Stop Mode

It is intuitively plausible to assume that the time delay measurement is directly causal, i.e. the laser pulse causes a photon event and the time delay between laser pulse and the subsequent photon event is then measured. However, most conventional TCSPC systems using TACs needed to give up this logical concept because of the typically high repetition rates of the excitation lasers: Since the time measurement circuit cannot know in advance whether there will be a fluorescence photon, it would have to start a time measurement upon each laser pulse. Considering that conventional TAC conversion times were often as long as .1 to 2 μs, any excitation rate in excess of .5 to 10 MHz would overrun the time measurement circuits. In fact they would most of the time be busy with conversions that never complete, because there is no photon event at all in most cycles. By reversing the start and stop signals in the time measurement, the conversion rates are only as high as the actual photon rates generated by the fluorescent sample. Historically there were (and had to be) only about 1 to 5 % of the excitation rate and could therefore be handled easily by the TAC / ADC. The consequence of this approach, however, is that the times measured are not those between laser pulse and corresponding photon event, but those between photon event and the next laser pulse (unless a long cable delay is inserted). This still works (by software data reversing) but is inconvenient in two ways:

1) Having to reverse the data leads to unpleasant relocation of the data displayed on a true time axis when the time resolution is changed.

2) Changing between slow and fast excitation sources requires reconnecting to different inputs.

The PicoHarp 330 and all other PicoQuant TCSPC devices are radically different in this respect, as they allow to work in forward start stop mode, even with fast lasers. This is facilitated by two design features:

1) Independent operation of the TDCs of all channels, and

2) a programmable divider in front of the sync input.

The divider allows to reduce the sync rate so that the TDCs and the subsequent processing pipeline can keep up with it. Internal logic determines the sync period and re-calculates the sync signals that were divided out. It should be noted that this only works with stable sync sources that provide a constant pulse-to-pulse period. Virtually all currently available fast laser sources meet this requirement within an error band of a few picoseconds. Note: for slow sync sources (< 1 MHz) the sync divider should not be used (set to None). Similarly, the divider must not be used in coincidence correlation measurements (or similar applications) when the sync input receives non-periodic (random) pulses from a photon detector. In summary: The PicoHarp 330 and all other PicoQuant TCSPC devices are designed to always work in forward start-stop mode.

Experimental Setup for Fluorescence Decay Measurements with TCSPC

The figure below shows a simple setup for fluorescence lifetime measurements using one input channel of the PicoHarp 330. The picosecond diode laser (PDL 800–B driver with attached laser head) is triggered by its internal oscillator (settable at 2.5, 5, 10, 20 and 40 MHz). The light pulses of typically <70 ps FWHM, are directed toward the sample cuvette via appropriate optics. A neutral density filter can be used to attenuate the light levels if necessary. Upon excitation, the fluorescent sample will emit light at a longer wavelength than that of the excitation light (Stokes shifted). The fluorescence light is filtered out from scattered excitation light by means of an op-

tical cut-off filter (other configurations may use a monochromator here). Then it is directed to the photon detector, again possibly via some appropriate collection optics, e.g., a microscope objective or a lens.

As a photon detector the PMT model 10721 from Hamamatsu is very convenient. It only needs a 5 V supply and has an instrument response width of <250 ps, allowing measuring lifetimes even much shorter than this via deconvolution. If a higher time resolution is required, the detector of choice is an MCP-PMT or a HPD. The electrical signal obtained from the detector (small negative pulses of typically −10 to −50 mV) is fed to the TCSPC electronics via a preamplifier (e.g., PAM 105 from PicoQuant). This gives pulses of −100 to −500 mV. The level trigger is then set for falling edge and e.g., 80 mV. All cabling is implemented with double shielded 50 Ω coax cable. If the detector is a SPAD module with TTL output then the input trigger level must be set to a positive voltage and rising edge. TTL signals must be attenuated (-10 dB) to avoid input damage and to reduce cross-talk. Some SPAD devices like the PDM series from MPD provide negative timing signals that can also be slightly attenuated (-10 dB) for lower cross-talk. The PDL Series laser drivers readily provide the electric sync signal needed for the photon arrival time measurement. This signal (a narrow negative pulse) is also fed to the TCSPC electronics via a high quality 50 Ω coax cable. When using the PicoHarp 330 in combination with the PDL 800–B or a similar PicoQuant laser driver, a 10 dB attenuator can be inserted directly at the sync output of the laser driver. This reduces cross-talk. when the detector signals are relatively weak.
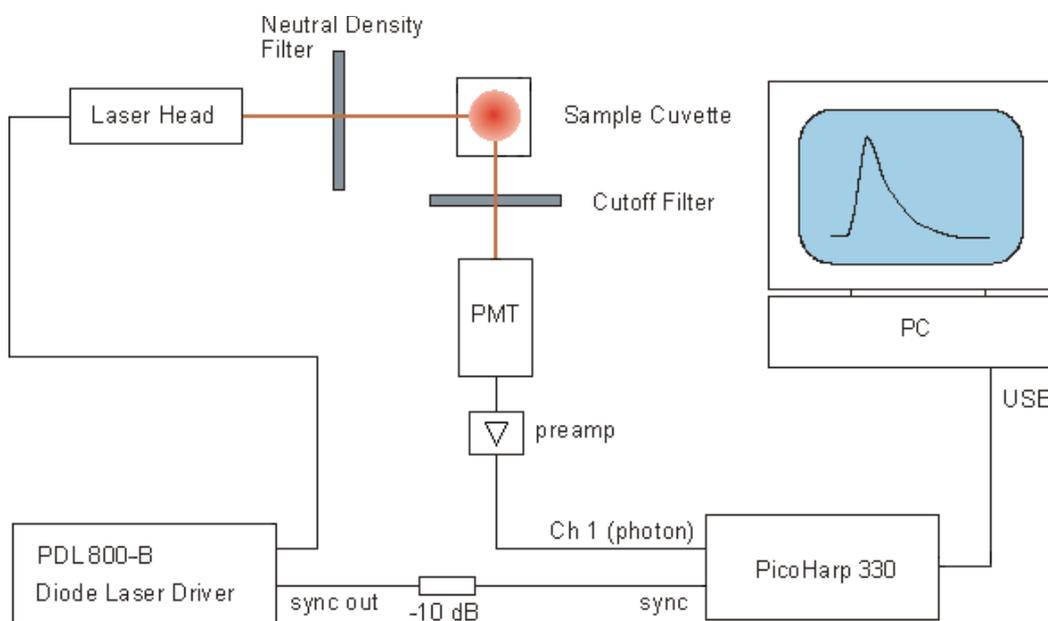
The Next figure shows TCSPC histograms obtained with this kind of setup. Excitation source was in this case a PDL 820 (PicoQuant) with a 438 nm laser head running at 20 MHz repetition rate. Detector was a PMA Hybrid 07 from PicoQuant with internal amplifier. The cut-off filter was in this case replaced by a monochromator set to let emission light pass at a band of 5 nm around 510 nm.

The narrower peak (blue curve) represents the system IRF, here dominated by laser and detector. The other curve (green) is the fluorescence decay from a solution of Coumarin 6 in ethanol, a fluorescent dye with relatively short fluorescence lifetime (~2.5 ns). The count rate was adjusted to <2 % of the laser rate to safely prevent pile-up. The plot in logarithmic scale shows the perfect mono-exponential nature of the decay curve, as one would expect for this dye. Note that this is obtained without a deconvolution of the IRF.

The approximate mono–exponential fluorescence lifetime can be obtained from a simple comparison of two points in the curve with counts in the ratio of $1 : {}^1/_e$ (e.g. 100 000 : 36 788). For a precise measurement a numerical exponential fit with IRF deconvolution (typically implemented as an iterative reconvolution) would have to be performed, resulting in slightly shorter lifetimes since the IRF broadens the decay. Indeed one can measure lifetimes significantly smaller than the IRF with this method. Additionally, the residues of the fit can then be used to assess the quality of the fit and thereby the reliability of the lifetime measurement. The "EasyTau 2" software package from PicoQuant provides this functionality.

E:\Coumarin6.phu - PicoHarp 330

File   Edit   View   Help

Coumarin 6 in EtOH, Exc. 438nm, Em. 510nm

Sync /cps	Input /cps	Histog. /cps	Total Count	Max. Count	at Time /ns	FWHM /ns
2,00e7	2,74e5	2,84e5	1,35e7	10000	2,933	1,832
	0,00e0	0,00e0	0,00e0	0	0,000	0,000

For help, press F1
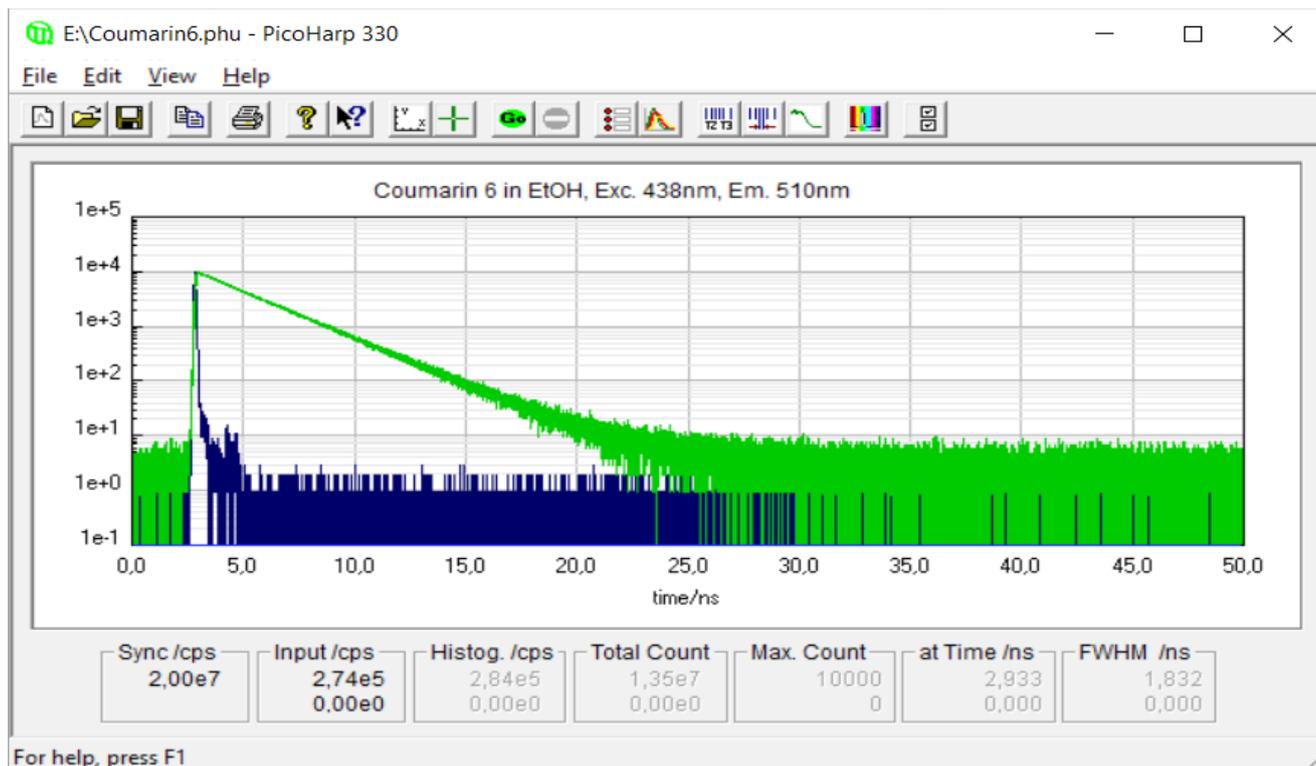
2.5. Further Reading

1. O'Connor, D.V.O., Phillips, D.:
   Time–correlated Single Photon Counting, Academic Press, London, 1984

2. Lakowicz, J. R.:
   Principles of Fluorescence Spectroscopy, 3rd Edition, Springer, New York, 2006

3. Kapusta, P., Wahl, M., Erdmann, R. (Eds.):
   Advanced Photon Counting - Applications, Methods, Instrumentation
   Springer Series on Fluorescence, Vol. 15, Springer International Publishing, 2015
   ISBN 978-3-319-15635-4

4. Ortmann U., Wahl M., Kapusta P.:
   Time-resolved fluorescence: Novel technical solutions.
   Springer Series on Fluorescence, Vol. 5, p.259-275, Springer International Publishing, 2008

5. Wahl M., Roehlicke T., Kulisch S., Rohilla S., Kraemer B., Hocke A.C.:
   Photon arrival time tagging with many channels, sub-nanosecond deadtime, very high throughput, and
   fiber optic remote synchronization. *Review of Scientific Instruments*, 91, 013108 (2020)

6. Wahl M., Rahn H.-J., Röhlicke T., Erdmann , R., Kell G., Ahlrichs, A., Kernbach, M., Schell, A.W.,
   Benson, O.: Integrated Multichannel Photon Timing Instrument with Very Short Dead Time and High
   Throughput. *Review of Scientific Instruments*, 84, 043102 (2013)

7. Wahl M., Rahn H.-J., Röhlicke T., Kell G., Nettels D., Hillger F., Schuler B., Erdmann R.:
   Scalable time-correlated photon counting system with multiple independent input channels.
   Review of Scientific Instruments, Vol.79, 123113 (2008)

8. Koberling F., Kraemer B., Buschmann V., Ruettinger S., Kapusta P., Patting M., Wahl M., Erdmann R.:
   Recent advances in photon coincidence measurements for photon antibunching and full correlation
   analysis. *Proceedings of SPIE*, Vol.7185, 71850Q (2009)

9. O'Connor, D.V.O., Ware, W.R., Andre, J.C.:
   Deconvolution of fluorescence decay curves. A critical comparison of techniques.
   J. Phys. Chem. 83, 1333–1343, 1979

10. Patting M., Wahl M., Kapusta P., Erdmann R.:
    Dead-time effects in TCSPC data analysis. *Proceedings of SPIE*, Vol.6583, 658307 (2007)

11. Patting M., Reisch, P., Sackrow, M., Dowler, R., Koenig, M., Wahl M.:
    Fluorescence decay data analysis correcting for detector pulse pile-up at very high count rates.
    Optical Engineering, 57(3), 031305 (2018). doi: 10.1117/1.OE.57.3.031305

12. Isbaner S., Karedla N., Ruhlandt D., Stein S.C., Chizhik A., Gregor I., Enderlein J.:
    Dead-time correction of fluorescence lifetime measurements and fluorescence lifetime imaging.
    Opt. Express 24(9):9429-45 (2016) doi: 10.1364/OE.24.009429

13. Bibliography listing all publications with work based on PicoQuant instruments
    http://www.picoquant.com/scientific/references

14. Web page of the PicoQuant TCSPC and time tagging devices
    https://www.picoquant.com/products/category/tcspc-and-time-tagging-modules

15. PicoQuant technical and application notes
    http://www.picoquant.com/scientific/technical-and-application-notes

3. Hardware and Software Installation

3.1. Scope

This chapter covers the hardware installation of the PicoHarp 330. Regarding software this manual describes solely **the standard PicoHarp 330 software** providing users of the instruments with an easy to use graphical interface. The PicoHarp 330 software runs on current x86-64 Windows PC platforms and also under x86-64 Linux with Wine (see section 8.4). It provides functions for setting measurement parameters, displaying measurement results, loading and saving of measurement parameters and decay curves, etc.

Important: The hardware and software of the predecessor product PicoHarp 300 is significantly different from that of the new PicoHarp 330. There is no software compatibility across the two products. Please do not confuse the two products. The PicoHarp 300 has its own manual and its own software. Here we cover only the PicoHarp 330.

In addition to the standard PicoHarp 330 software there are a variety of other relevant **software items that are not covered by this manual**:

PH330Lib is a programming library enabling users to write custom data acquisition programs for the PicoHarp 330 in virtually all popular programming environments for Microsoft Windows. There is also a version for Linux which is fully compatible with that for Windows so that applications can easily be ported across the two operating systems. The programming library is not covered here in this manual. Please see the separate installation and documentation files (distinct for Windows and Linux) provided in the distribution pack available via download from the PicoHarp 330 product pages on the Web.

A relatively advanced **high-level API package for Python called "snAPI"** is also available via GitHub. (Windows only). It readily provides data collection and file writing methods as well as many real-time analysis methods such as intensity and coincidence time traces, FCS and $g^{(2)}$ correlation.

Further items of PicoQuant software of potential interest are the EasyTau 2 software, the SymPhoTime 64 software, and the QuCoa software. These allow highly professional, application focused use of the devices in the contexts of time-resolved spectroscopy, microscopy, and quantum optical research, respectively. Please see the PicoQuant website for more information on these items.

3.2. What's New in this Version

Software version 2.0.0.0 is the second release for the PicoHarp 330 system which primarily adresses support for the new four channel device PicoHarp 330 4P. It also provides an extended debug facility where device internal states of the FPGA system can now be saved and submitted for support. There were also some bugfixes in the user interface (control panel, axis panel, general settings dialog) and in the real-time correlator. The software is now allowing to open PicoHarp 330 .PTU files (T3 mode only) from SymPhotime 64 and snAPI. The PicoHarp 330 software is now distributed via download only. See https://www.picoquant.com/downloads and look in the category TCSPC and Time Tagging Electronics. Do not confuse the software for PicoHarp 300 and that for PicoHarp 330, what you need is the latter.

Users of other PicoQuant TCSPC systems will find the user interface very familiar. Compared to such earlier products the file format remains conceptionally unchanged, except that in case of extended features (such as the programmable input configuration) the file header now contains the corresponding settings. Thanks to our tagged file format (.PTU) this change is transparent for existing software and PicoQuant's data analysis software such as snAPI, SymPhoTime 64, QuCoa and EasyTau 2 remain compatible. Nevertheless it is recommended to upgrade such software to their latest version.

3.3. General Installation Notes

When handling the PicoHarp 330 system, make sure to avoid electrostatic discharges, especially when connecting cables. Before connecting any signals, carefully study the maximum ratings given in section 8.3.1.

Note that the other PicoQuant TCSPC devices have their own software and their own manuals. The PicoHarp 330 software does not work with these devices and this manual does not relate to them in any meaningful way. Most importantly, note that the historic PicoHarp 300 is a different device and needs different software, and vice versa the PicoHarp 330 cannot work with the historic PicoHarp software.

The PicoHarp 330 software version 2.0.0.0 is compatible with Windows[1] 10 and 11 as of August 2024. Windows 7 and 8 may still work but are no longer tested and supported. Consider the security risks of using an outdated operating system. Future Windows versions are likely to work but obviously cannot be tested before they are released, and are therefore not formally supported.

In order to use the PicoHarp 330 software with the PicoHarp 330 hardware, a compatible device driver must be installed. The PicoHarp 330 is a 'plug and play' device, meaning that necessary resources and drivers are allocated automatically by the operating system. The software setup will conveniently cater for the installation of the driver.

On some Windows installations you may need administrator status to perform software setup and de-installation. For the driver installation, it is needed in any case. Installing as administrator has the benefit that you can install the software for all users on that computer even if they have limited access rights. The PicoHarp 330 software will maintain individual settings for each user in the Windows registry. Note that the PicoHarp 330 software does not support running in multiple concurrent user sessions.

3.4. Software Installation

Starting with version 2.0 the PicoHarp 330 setup files are no longer provided on physical media. Please download the software pack from the PicoQuant website at https://www.picoquant.com/downloads. On the download page scroll down to the category `TCSPC and Time Tagging Electronics` and select `PicoHarp 330 – Operation Software`. The download will be packed in a ZIP-File. Please unzip this file to a temporary hard disk location of your choice. You will need administrator status to perform the setup for all users. For the device driver installation you need it in any case.

For access to the PicoHarp 330 hardware, a device driver must be installed. The PicoHarp 330 is a 'plug and play' device, this means that the necessary resources and drivers are allocated automatically by the operating system. Windows will recognize when such a device is connected and tries to install the appropriate driver. It is recommended to perform the software setup before connecting the device because then Windows will find the driver readily installed. If you skipped that step, Windows may report that it could not install the driver. In that case you can still run the software setup as described below.

After you downloaded the zip archive and unpacked the setup files to a hard disk location of your choice, open that location. You will find a folder strucure with multiple items where you must navigate to the sub-folder `PicoHarp 330 v.X.X.X.X\installation\`. The installer program file containing the software including driver, documentation and file demos is named `setup.exe`.
Run `setup.exe`. The setup program will guide you through the installation process step by step.

Setup

When asked for a destination folder for the new software, please accept the default path or select another according to your program storage policies. This is where the PicoHarp 330 application files will be installed. To avoid confusion, make sure not to specify the path of an older PicoHarp 330 version that you have not uninstalled or that of any other program on your PC.

The default location is: `\Program Files\PicoQuant\PicoHarp330v20`.

Setup will also create a dedicated "program folder" for the new PicoHarp 330 software that will later appear in the Start Menu. You can accept the default folder name or select another according to your own naming policies. However, you should make sure not to specify the folder name of an older PicoHarp 330 version that you have not uninstalled nor the dedicated folder of any other program.

In the chosen destination folder the installer will also create a subdirectory `\filedemo` which contains demo source code for access to PicoHarp 330 data files in various programming languages. Furthermore, another folder `\sampledata` will be created with samples of PicoHarp 330 data files. Other necessary files such as setup information and the device driver will be installed in the standard places in your Windows directory tree.

The setup program will also optionally install the device driver and a *File Info* shell extension that you can use to inspect individual header items of a `*.ptu` or `*.phu` file. This includes the measurement mode. Just right-click on the file in Windows explorer and select *Properties*. Then look at the tab *PQ File Info* and the tab *PQ File Comment*.

After the installation the PicoHarp 330 software should be available in the Windows Start Menu under the custom folder name you chose during setup. If you accepted the default then it will appear under
Programs | PicoQuant – PicoHarp 330 v2.0.

---

1   Trademark of Microsoft Inc.

3.5.   Hardware Installation

3.5.1.   Electrical Connection

Make sure to *prevent electrostatic discharge* when unpacking and handling your PicoHarp 330, especially when connecting cables. Note that some detectors operate with high voltage that may discharge through the signal cable. Make sure such detectors are switched off and fully discharged before connecting them.

The PicoHarp 330 requires power from a regular 100...240V AC line. The corresponding IEC socket at the back of the housing also holds the power switch and two fuses. The PicoHarp 330 has no dedicated power indicator, however, once power is connected and switched on you should see some of the status LEDs at the front light up. See section 8.3.3 for their interpretation.

Also at the back of the housing there is a grounding terminal that can be used to make a connection to the grounding line of your lab. Ask a qualified electrician when in doubt what and where this is. This grounding is not necessary for electrical safety, however, it may in some situations help to reduce noise pickup. In order to be effective the grounding should be of low impedance, i.e. using thick wires, less than 3 m long, ideally as short as possible.

The PicoHarp 330 has a USB 3.0 super speed interface which is backwards compatible with USB 2.0. In order to obtain the maximum throughput it must be connected through USB 3.0 or higher. Consult your PC manual as to whether and where it provides high speed USB 2.0 or super speed USB 3.x connectors. The latter can usually be identified by their blue color. The PicoHarp 330 will not work with a USB 1.x connection. However, all current PCs should readily provide USB 3.0 connectivity. Recent PCs typically provide USB 3.1 or 3.2 connections which are backward compatible with USB 3.0 and should be fine for running the PicoHarp 330. If the PC has no USB 3.x ports you can install a USB 3.x adapter card but this may not provide the same throughput as an on-board controller.

Always use a quality USB cable rated for USB 3.0 or higher speed. The cable length must not exceed 5 meters (~16 ft). For best reliability we recommended to use the provided cable of 3 meters length. Note that the USB specification does not allow cable extensions other than dedicated active extension cables or hubs. The Pico-Harp 330 should work flawlessly through suitable USB hubs. This is also a valid way of extending the maximum cable length. After a hub, another cable is formally allowed, the same length restrictions as mentioned above will apply. Note, however, that hubs may lower the data throughput. Also for throughput reasons it is recommended not to connect other devices with high bandwidth requirements to the same hub.

The steps below describe how to connect the signals for your experiment. The inputs for the photon detector(s) and the sync signal are SMA connectors located on the front panel of the PicoHarp 330. They are labeled SYNC, CH1, CH2, … up to CH4, dependent on which hardware model you have. The sync input is typically used for synchronization with a laser, while CH1 … CH4 are detector signal inputs. The sync input can also be used for detectors if a laser sync is not required, e.g., in antibunching measurements or in some generic time tagging applications. All inputs are terminated with 50 Ohms internally. Use quality 50 Ohms coax cables with appropriate connectors. For interfacing to BNC connectors, use standard adapters. Carefully screw on the SMA connectors for sync and detector(s) until they are hand-tight. Do not use wrenches. Your PicoHarp 330 ships with a dedicated SMA connector tool that allows convenient handling in limited space without risk of over-tight-ening. Note that some detectors (notably PMTs) operate with high voltages that may discharge through the signal cable. Make sure such detectors are switched off and fully discharged before connecting or disconnecting them.

In case of time resolved fluorescence experiments with a pulsed excitation source, the sync signal must be connected to the sync input and the detector signal(s) must be connected to the detector input(s). If coincidence correlation experiments between two (or more) detector signals are to be carried out, you need to decide whether you will be using histogramming or TTTR modes T2/T3. In the case of histogramming and T3 mode connect one detector to the sync input and one or more to the detector inputs. Histogramming and T3 timing will always be with respect to the sync input. In T2 mode it is possible to determine the relative timing between all inputs but this requires other software and/or off–line data analysis (see section 5.3).

Connect the other signal cable ends to the appropriate signal sources (50 Ω) in your experimental setup. The inputs of the PicoHarp 330 accept negative pulses with peak values down to -2 V and positive pulses up to +2.4 V. When an input is configured as a CFD then the detector pulse must be negative and the software allows setting the CFD level and CFD zero crossing. When an input is configured as an edge trigger the software allows the selection of the trigger edge (rising=1 or falling=0) and the trigger level. Ideally, all inputs should be operated with similar pulse amplitudes to minimize cross-talk. The optimum amplitude range is 200 to 500 mV. Below this range you may pick up noise, above there may be increased cross-talk. Most PMT and MCP detectors will require a pre–amplifier to reach a sufficient signal level. Weak PMT detectors should be connected through

a 20 dB high speed pre–amplifier. MCP–PMT detectors should be connected through an amplifier with slightly higher gain. TTL–SPAD–detectors must be connected through an attenuator or an attenuating inverter (Pico-Quant SIA 400). **Never connect TTL signals to a PicoHarp 330 directly, as this may cause damage.** When detectors with small signals are being used in combination with laser drivers from PicoQuant's PDL Series, the sync pulses from the laser driver should be attenuated by 10 dB to fall into the optimum range for smallest cross-talk. Similar attenuation is recommended on the detector signals when detectors with NIM output are used. Suitable attenuator and amplifier devices are available from PicoQuant.

IMPORTANT: Switch the high voltage supply of PMTs off and allow their electrodes to discharge before connecting them. Their high voltage charge may damage the pre–amplifier. Observe the allowed maximum ratings for the input signal levels. Above these levels hardware damage will occur. If you are not sure what signals your setup delivers, use a fast oscilloscope (1 GHz bandwidth or better) to check the signal level and shape before connecting them to the PicoHarp 330. All signals should have fast rise times of no more than a few ns. Slower signals may degrade timing accuracy.

Do not connect anything other than dedicated hardware to the Sub-D multi-pin connector (control port). It is provided for hardware expansion (notably experiment control) and must not be used otherwise. See section 8.3.2 for pin assignments. It is recommended to start the instrument setup without anything connected to the control port.

3.6.　Installation Troubleshooting

After completion of the software setup and connecting the PicoHarp 330 the device should be listed in the Windows Device Manager. Right click on the Windows start button and select *Device Manager* to check if the device is free of conflicts and / or if the device driver is installed correctly. Under *USB Devices* look for a device named *PicoHarp 330* and inspect its *Properties*.

A common source of problems is the shutdown behavior of Windows 10 and 11. It does not fully shut down by default but goes only into a state similar to hibernation in order to re-start more quickly. When new hardware is installed this can cause problems. If you missed this during hardware installation, consider it at least when problems arise. In order to fully shut down Windows hold the shift key while clicking the shutdown button or run the command `shutdown /s /t 0` from the command prompt. It is also possible to permanently configure Windows for proper shutdown via the "power options" dialog.

If the PicoHarp 330 driver is not installed or needs updating you can install it manually. When prompted to search for the driver, direct the driver wizard to search the location where you unpacked the downloaded software pack. The simpler solution ist to just repeat the software installation process.

If things are not working as expected you can also use the Windows system information facilities (*Start > Run > msinfo32).* In the System Information utility inspect *Software environment > System Drivers* to check if the Pico-Harp 330 device driver PQCYUSB.SYS is correctly installed.

You can also repeat the software installation if necessary. To do so, first uninstall the software and repeat the setup procedure. Make sure the software is not installed in multiple places. If this does not resolve the problem, try a different computer. If problems persist, see section 7 for support.

Should the software appear sluggish during measurements it may be because the USB connection is running only at USB 2.0 speed. The USB LED on the front panel of the PicoHarp 330 will then show yellow. At proper USB 3.0 superspeed it will light up green. Check your USB cable and try other USB ports of the PC to get this fixed.

3.7.　Uninstalling the Software

Before uninstalling the PicoHarp 330 software you should back up all PicoHarp 330 data files you might have created in the installation directory.

Do not manually delete any program files from the installation folder as this may prevent a clean uninstall process.

To uninstall the PicoHarp 330 software from your PC you may need administrator rights (depending on Windows version and security settings). From the list of installed applications select *PicoQuant – PicoHarp 330 vx.x* for un–installation. This will remove all files that were installed by the PicoHarp 330 setup program but not the user data that may have been stored. If there was user data in the program folders or any subfolders, these will not be deleted by the uninstall program. If intended you need to delete these files or folders manually. Nevertheless, it is recommended to back up valuable measurement data before uninstalling the software.

Note that uninstallation of the data acquisition software does not uninstall the device driver. This is because other software may still need it. You can delete the driver software from within Device Manager.

Note also that un–installation of the data acquisition software does not automatically uninstall the PQ File Info shell extension. It can be un-installed as a separate item through the standard Windows mechanisms under *Apps and Features*.

4. Software Overview

4.1. Starting the PicoHarp 330 Software

After correct installation the Windows Start Menu contains a shortcut to the PicoHarp 330 software. To start the PicoHarp 330 software select *PicoQuant – PicoHarp 330 vX.X – PicoHarp 330*. Note that after switching the device on, you need to allow a warm–up period of about 20 minutes before using the instrument for serious measurements. You can use this time for set–up and preliminary measurements.

If the PicoHarp 330 software cannot find a PicoHarp 330 device (or if there are driver problems) it will display a notification message, but it will still start. However, device dependent toolbar buttons and functions of the pro-gram will then be disabled. This allows you to use the software without the PicoHarp 330 hardware, e.g., to view or print files on another computer.

It is possible to use up to eight PicoHarp 330 devices on one PC. If multiple devices are installed then the first instance of the software will connect to the first device, the second to the second device and so on. Each of the PicoHarp 330 software instances displays the serial number of the device it uses. An instance that does not find an unused device will open as a file viewer only.

Note that the various instances of the PicoHarp 330 software will be running completely independent of each other. If your application requires some kind of joint action of multiple PicoHarp 330 devices then you must de-sign your own software based on the PicoHarp 330 programming library PH330Lib (see separate manual). If your objective is to combine multiple devices to obtain more detector channels you may need to synchronize the clocks of the devices. You also need to consider that it is not possible to prevent Windows from introducing un-predictable delays in communication with the hardware. The latter makes it impossible to, e.g., start measure-ments on multiple devices at the exact same time. See the PH330Lib manual and the demos for partial solu-tions.

If the PicoHarp 330 is correctly installed and there are still hardware related errors, you can use the Windows device manager for troubleshooting (see the corresponding section above). If problems cannot be resolved, see section 7 for support. If possible, try using the device with another computer first.

For regular use of the PicoHarp 330 software you may want to use the icon the software installer created on your Windows desktop. You can also start up the PicoHarp 330 software directly from a PicoHarp 330 data file by double–clicking on the file or dragging it onto the PicoHarp 330 icon. Note that this may not work as expected if you have other PicoQuant TCSPC devices and associated software installed on the same computer. The file name extensions may then be assigned to that other software.

4.2. The Main Window

The PicoHarp 330 software provides a measurement control interface to the PicoHarp 330 hardware and an on-line histogram display. Most prominently the PicoHarp 330 main window accommodates the histogram display area.

Above the display area is the toolbar (see section 4.3). Here you can access frequently used commands by sim-ple mouse clicks. Next to the toolbar buttons you see the serial number of the device in use. This can be useful if you are running multiple instances. Above the toolbar is the menu bar with additional commands. At the bot-tom of the histogram display area is a set of 'panel meters' showing count rates, count sums, and histogram peak characteristics. These will be updated continuously, some only when a measurement is in progress. The panel meters can be enlarged by double-clicking them, which is useful when performing optical alignment or similar tasks when the PC monitor is some distance away.

In the top center of the display area a title line is shown. This can be double-clicked to edit the title. When editing the title, note that only the first line will appear in the display. The remaining lines are meant to be used as a file comment. All lines will be stored with the file data, with a maximum of 256 characters.

The main window is resizeable and the actual histogram display will adapt its size accordingly. If you make the window smaller than the minimum histogram display, two scrollbars will permit access to hidden window areas. Note that the actual size of the main window depends on the system font selected. It will scale to the size of the current system font. Screen resolutions below 800x600 are not suitable for serious work with the software.

Note that the position and size of the main window on the screen will be stored in the Windows registry and re‐trieved upon the next program start. The registry settings are kept separately for each user, provided he / she is logged on with a personal user account. Consult section 6.1 for further main window command descriptions. Toolbar, Menus, panel meters etc. will be explained in the next sections.

At the very bottom of the main window there is a status bar. The leftmost area of the status bar describes ac‐tions of menu items as you navigate through menus. Similarly it shows messages that describe the actions of toolbar buttons. The second status bar area from the left shows the current measurement status of the Pico‐Harp 330. The rightmost area of the status bar indicates if the `<Caps>` and `<NumLock>` keys are latched down.

When the PicoHarp 330 software is running with functional hardware it continuously collects informa‐tion about the input signals and the current acquisition settings. If these settings together with the in‐put rates indicate possible errors, the software will display a warning icon in the status bar. The warning icon can be clicked to review the list of current warnings together with a brief explanation (see also section 8.1).

4.3. The Toolbar

The toolbar is displayed across the top of the PicoHarp 330 software main window, below the menu bar. The toolbar provides quick mouse access to frequently used commands and tools. Note that some buttons may be grayed out (disabled) depending on the state of the hardware.

To hide or display the toolbar, choose Toolbar from the View menu (`<Alt>+V T`). The following table explains the individual buttons.

Click…	to…
	open a blank histogram with default control panel settings
	open an existing histogram file. Displays the *Open* dialog box, in which you can locate and open the desired file.
	save the current histogram data with its current name. If you have not named the file, the *Save As…* dialog box is displayed.
	copy the currently displayed curves to the clipboard (ASCII export).
	print the currently displayed histogram curves.
	display the *About…* window. There you can obtain the versions of your PicoHarp 330 software / hardware as well as links for updates and support.
	activate context sensitive help.
	launch the axis panel.
	launch the data cursor dialog.
	start measurement based on current PicoHarp 330 control panel settings
	stop measurement and histogram accumulation.
	launch the PicoHarp 330 control panel.
	launch the trace mapping dialog.
	launch the TTTR mode dialog.
	launch the filtered TTTR mode dialog.
	Launch the TTTR mode real–time correlator dialog.
	launch the configuration dialog for SEQ mode.
	launch the general settings dialog.

4.4.   The Control Panel

The PicoHarp 330 control panel is a dialog box for setting the parameters for hardware adjustment and data ac-quisition. It is implemented as a 'non–modal' dialog box, i.e. it does not have to be closed before the main win-dow can continue to operate. This way you can make changes to your settings in the control panel and watch their effect on a running measurement in the main window immediately. Nevertheless, you may close the control panel and restore it at any time by clicking the control panel button on the toolbar or pressing `<Alt>+C`.
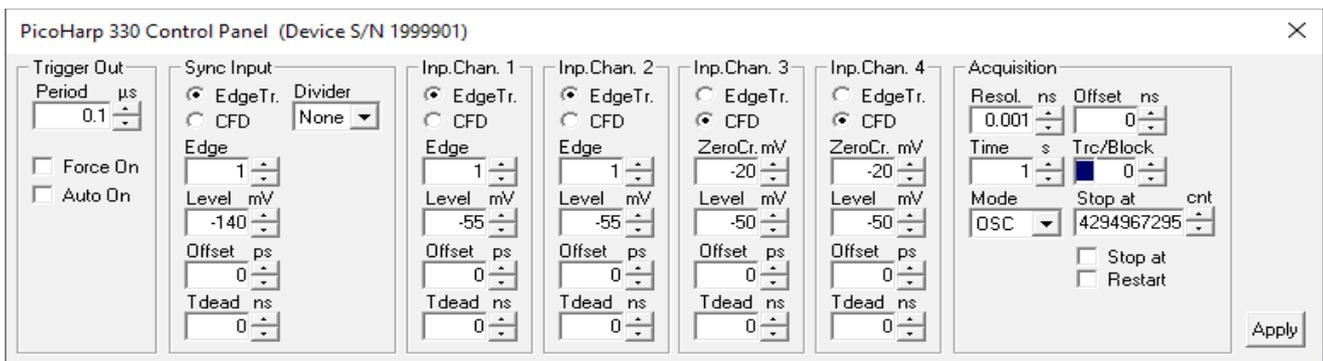
The control panel consists of several groups of edit boxes and other controls for related parameters. These groups and their respective controls are:

Trigger Out

Period	edit box and spin control	Enter the desired trigger output perion in μs.
Force On	tick box	Tick to force the output always on.
Auto On	tick box	Tick to have the output switch on while measuring.

Sync Input

EdgeTr.	radio button	Select to switch the input to Edge Trigger mode.
CFD	radio button	Select to switch the input to CFD mode.

when above EdgeTr. is selected:

Edge	edit box and spin control	Enter the desired trigger edge (0=falling, 1=rising).
Level	edit box and spin control	Enter the desired trigger level in mV.

when above CFD is selected:

ZeroCr.	edit box and spin control	Enter the desired zero-cross level in mV.
Level	edit box and spin control	Enter the desired discriminator level in mV.
Offset	edit box and spin control	Enter the desired time offset ("delay") in ps.
Tdead	edit box and spin control	Enter the desired dead time of the sync input.
Divider	dropdown selection box	Select the desired sync divider.

Inp. Chan 1 ... Inp. Chan 4

EdgeTr.	radio button	Select to switch the input to Edge Trigger mode.
CFD	radio button	Select to switch the input to CFD mode.

when above EdgeTr. is selected:

Edge	edit box and spin control	Enter the desired trigger edge (0=falling, 1=rising).
Level	edit box and spin control	Enter the desired trigger level in mV.

when above CFD is selected:

ZeroCr.	edit box and spin control	Enter the desired zero-cross level in mV.
Level	edit box and spin control	Enter the desired trigger level in mV.
Offset	edit box and spin control	Enter the desired time offset ("delay") in ps.
Tdead	edit box and spin control	Enter the desired dead time of this channel.

Controls for input channels that are not available in the device being used will remain disabled (gray).

Acquisition

Resolution	edit box and spin control	Enter the desired time resolution in ns.
Time	edit box and spin control	Enter the desired acquisition time in s.
Mode	drop down selection box	Select the desired measurement mode.
Offset	edit box and spin control	Enter the desired histogram offset in ns.
Trc/Block	trace color indicator,	Click to open the Trace Mapping Dialog.
	edit box and spin control	Select the next data block to be used.
Stop at	edit box and spin control	Enter the desired count level to stop at.
Stop at	check box	Tick to let the measurement stop at the given level.
Restart	check box	Tick to make the measurement re-start automatically.

Edit boxes are for keyboard entry. The values must be confirmed with either the `<Enter>` key or the *Apply* button. The spin controls can be used to increment or decrement the value in the edit box. In this case the changes take effect immediately without need for hitting `<Enter>` or clicking *Apply*. Check boxes have their denoted effect when the tick is shown. They can be toggled with a mouse click. Groups of radio buttons are like check boxes but mutually exclusive.

Note that the settings of the control panel as well as the positions of control panel and main window on the screen will be stored in the Windows registry and retrieved during the next program start. The registry settings are stored separately per user. When a PicoHarp 330 data file is loaded, the control panel settings will change to reflect the settings stored in that file.

The individual control panel items are explained further in section 6.4 in the Controls and Commands Reference.

4.5.   The Axis Panel

The axis settings panel is a dialog box for setting the axis range for the histogram display in the main window. It is implemented as a 'non–modal' dialog box, i.e. it does not have to be closed before the main window can continue its operation. This way you can make changes in the axis panel and watch their effect in the main window immediately. Nevertheless, you may close the axis panel and restore it at any time by clicking the axis panel button on the toolbar or pressing `<Alt>+A`. The panel will also open if you double–click the axes in the main window.
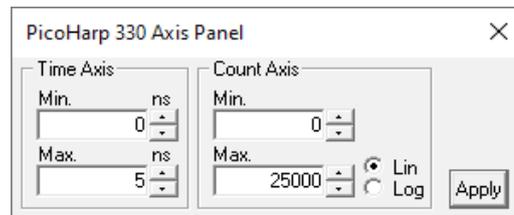
The axis panel consists of two groups containing edit boxes and other controls for related parameters. These groups and their respective controls are:

Time Axis

Min.	edit box and spin control	Enter the time axis lower bound in ns.
Max.	edit box and spin control	Enter the time axis upper bound in ns.

Count Axis

|		
Min.	edit box and spin control	Enter the count axis lower bound.
Max.	edit box and spin control	Enter the count axis upper bound.
Lin	radio button	Select linear count axis scaling.
Log	radio button	Select logarithmic count axis scaling.

The edit boxes are for keyboard entry. The values must be confirmed with either the `<Enter>`–key or the *Apply*–button. The spin controls can be used to increment or decrement the values in the edit box. In this case the changes take effect immediately without need for hitting `<Enter>` or clicking *Apply*. The mouse wheel can be used for fast spins as long as the cursor is in the corresponding edit box. Check boxes have their denoted effect when the tick is shown. They can be toggled with a mouse click. Groups of radio buttons are like check boxes but mutually exclusive.

Note that the settings of the axis panel as well as the positions of the panel on the screen will be stored in the Windows registry and retrieved during the next program start. The registry settings are stored on a per user basis.

The individual axis panel items are discussed in section 6.5. "Axis Panel" in the Controls and Commands Reference.

Hint: to begin, use logarithmic logarithmic count axis scaling to make sure even weak signals can be seen.

4.6.   The Trace Mapping Dialog

The PicoHarp 330 software can record and store histograms in up to 512 data blocks in memory and files. Out of these, up to 16 curves can be displayed at the same time. It may seem odd that only 16 traces can be displayed. However, it is simply getting too cluttered when too many traces are drawn on the same screen. The limitation to 16 is still allowing distinguishable colors and keeping the view reasonably tidy. The *Trace Mapping* dialog is used to select the curves to display. It also allows to view curve details and to delete curves. You can use the *Trace Map* button on the Toolbar or click the curve color indicator in the control panel to launch the dialog.
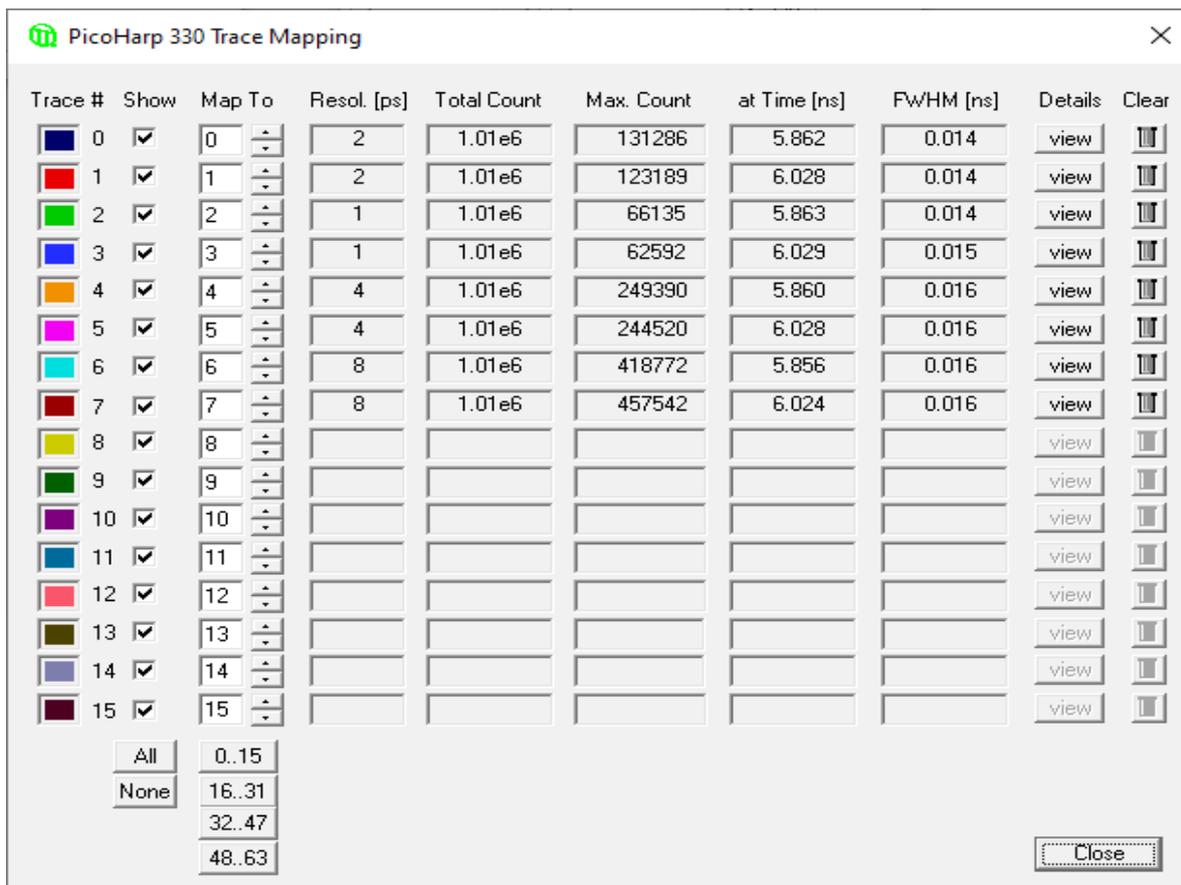
In the *Trace Mapping* dialog you can tick the individual boxes '*Show*' to display a curve. You can also select the index of the memory block you wish to map the individual display curves to.

The dialog also provides some statistics on each curve (central matrix of figures). *Resolution* is the bin width of the histogram in picoseconds. Next is the *Max. Count*, the count in the highest point of the curve. The column *at Time* shows the time corresponding to the *Max. Count* bin. Leftmost there is the Full Width Half Maximum (*FWHM*) of the curve peak (usually meaningful only for IRF traces).

There are also some buttons for frequently required actions: The button '*view*' can be clicked to see more detailed curve information such as time of recording, acquisition settings and count rates. The button *All* can be clicked to tick all traces as shown, the button *None* does the opposite. The button *0..15* can be clicked to set the default mapping of trace 0..15 to block 0..15. Similarly, the buttons below allow mapping to the subsequent groups of 16 blocks. This allows quick access to the upper traces when the device has more than 16 channels. The *Clear* buttons (trash cans) can be clicked to delete the contents of individual blocks.

Note that the Trace Mapping dialog is non–modal. This means the dialog can remain open while a measurement is in progress, so that adjustments can be made under immediate visual control, similar to the operation of the control panel. Note also that a measurement can be running in a block that is not mapped or shown.

The individual trace mapping panel items are explained further in the section 6.6. "Trace Mapping" in the Controls and Commands Reference.

4.7. Other Dialogs

In order to keep this manual readable, the dialogs described here in the overview chapter are limited to the most important ones the reader should know about before starting practical work with the software. Additional dialogs will be described implicitly in the following sections in the context of specific measurement tasks. For information on all other dialogs the user is kindly referred to the controls and commands reference (section 6) or consult the on–line help facility of the software. Pressing F1 in an active dialog will open a corresponding help page.

5. Specific Measurement Tasks

5.1. Setting Up the Input Channels

This section provides help and instructions for the first basic steps of setting up the instrument. However, if you are running TCSPC measurements for the first time we strongly recommend you read the primer on TCSPC in section 2 first. Also consider the literature listed there.

In order to acquire any data, the input channels and the sync input of the PicoHarp 330 must be set to match their electrical input signals. The PicoHarp 330's input channels are all designed identically. The PicoHarp 330 software allows the configuration of all inputs to either use a CFD or a simple comparator (edge trigger). Because of its delay element a CFD requires time to make its "decision" and this time is larger than the PicoHarp 330's internal TDC dead time. Since the short dead time is a precious feature when using high speed detectors, the use of a CFD would spoil the benefit. Indeed many modern detectors, notably SNSPDs, have very steep signal edges that do not require a CFD. A simple programmable comparator is actually beneficial here. Just like the detector signal, the sync signal must be made available to the timing circuitry. Since the sync pulses are usually of well defined amplitude and shape, an edge trigger is sufficient to accommodate most sync sources. In order to allow the best match for the given setup the PicoHarp 330 permits both configurations. When an input is configured to use the programmable edge trigger it will allow the selection of the trigger edge (rising=1, falling=0) and the trigger level in mV. When an input is configured as a CFD it will allow the selection of the zero cross and the trigger level, both in mV. For specification details see section 8.3.1 and take note of the maximum ratings.

In case of coincidence correlation experiments using two or more detectors, the input channels are typically used for one detector each and data is collected in T2 mode. In that case the sync input can (but need not) be used for a detector as well. If a quick visualization of coincidence counts is required, it is also possible to use histogramming mode. In that case, one detector is connected to the sync input. Coincidence histograms will be collected for each input channel with respect to the sync input. In time–resolved fluorescence measurements with a pulsed excitation source (typically a laser) the sync input receives a sync signal from the laser. Here we focus on the latter, more common case. Perform the following steps to set up the signal inputs.

Using the Panel Meters

At the bottom of the main window you find a set of panel meters. These are very important during set–up. The meters showing units of cps (counts per second) in their title are rate meters. The leftmost rate meter shows the sync input rate. The next meter shows the other channel input rates. The other meters show the histogramming rates, the total count in the histograms, the maximum (peak) counts and the position of the maxima for the present input channels. The following example uses the same generator for all inputs, this is why the rates are all identical.

Sync /cps	Input /cps	Histog. /cps	Total Count	Max. Count	at Time /ns	FWHM /ns
1.01e6	1.01e6	1.01e6	1.01e6	70289	5.871	0.005
	1.01e6	1.01e6	1.01e6	68260	5.989	0.006
	1.01e6	1.01e6	1.01e6	68050	5.980	0.006
	1.01e6	1.01e6	1.01e6	69169	5.901	0.006

Note that the rate meters use a fixed gate time of 100 ms. Their accuracy at low rates is therefore limited. They really only serve as a quick means of diagnostics and should not be used to obtain definitive measurement results. Note also that you can double–click all the rate meters to enlarge them as separate windows. This is useful for optical alignment work if the PC screen is some distance away from your setup.

Setting up the Sync Input

For typical fluorescence decay measurements, the PicoHarp 330 needs an electrical sync signal from the light source. The PDL Series of diode lasers from PicoQuant provides this signal directly. If the laser does not provide an appropriate electrical sync signal (e.g.,some Ti:Sa lasers), a sync detector (photo diode) such as the TDA 200 must be used. The sync signal must consist of pulses with steepness and amplitude matching the specifications of the PicoHarp 330. For example, a NIM type signal is appropriate. This is a steep negative pulse

(0.5 to 10 ns wide, active edge falling) of typically −800 mV into 50 Ω. For such pulses one would configure the sync input as an edge trigger. As the trigger edge must be set to the leading edge one would select falling (=0).

The PicoHarp 330 can actually handle over ±2 V but large amplitudes may cause excess interference and cross-talk. between the inputs. Amplitudes around 200 to 500 mV (on all inputs) are best in terms of timing accuracy and lowest histogram ripple due to crosstalk. It may therefore be advantageous to attenuate NIM pulses by 10 or 15 dB. Lowest cross-talk. is typically achieved by using signals of similar amplitude on all inputs. SMA in-line attenuators of suitable bandwidth can be used to adjust this. The trigger level is adjustable for optimum timing accuracy. Initially you should set it to half of the sync pulse amplitude. Later it can be fine-tuned empirically.

Unless you are sure what kind of signal your sync source delivers, use a fast oscilloscope (50 Ω input) to check the pulse shape, polarity and amplitude. The leading edge should be steep (ideally 2 ns rise ∕ fall time or faster), there should also be no excessive ringing. The pulse width should be at least 0.4 ns, the upper limit is not critical.

If the signal is satisfactory, connect the source to the sync input and start the PicoHarp 330 software if it is not yet running. A detector signal (CH1, CH2, ...) is not required at this point but it does no harm if it is also connected. Open the PicoHarp 330 control panel. Leftmost, there is a group of controls for the sync input. Find the edit box and spin control for the trigger level. The level should be set to a value around half of the amplitude of the sync pulses. Next, set the appropriate signal edge (typically the leading edge of your signal). The code 0 means falling, 1 means rising. Then look at the sync divider. It must be set so that the sync rate divided by the shown divider value remains under 81 MHz. For all slower sources it should be set to None. *Tdead* is a programmable dead time for suppression of afterpulsing artefacts of some detectors. For the sync input it is rarely needed. Leave it at 0 for now.

If the sync source is active, the sync rate will now be displayed at the leftmost rate meter in the PicoHarp 330 main window. Note that the sync rate meter internally corrects for the chosen divider setting, so that the meter always shows the undivided input rate. The meter display should therefore match the rate delivered from the source. Note that meaningful use of the divider requires periodic sync signals. The rate meter will be refreshed every 0.1 to 1 seconds, as determined by the value in the General Settings Dialog. The sync rate should be displayed very accurately unless it is very low. Large fluctuations or occasional zeros indicate an incorrect discriminator level setting or an unstable sync signal. Try varying the discriminator level to obtain a stable sync rate display. If the rate is stable and at the expected value, you can proceed to set up the other inputs. A last fine tuning of the trigger level can be done when the detector inputs are up and running. Note that it may be impossible to get a stable sync rate reading at very low sync rates. This is because the rate counter uses a prescaler and a gate time of 100 ms and if the signal period is too low the readings will fluctuate. You will then have to set the level based on what you know about the pulse amplitude and verify it in an actual measurement.

Setting up the Photon Detector Inputs (Channel 1…N)

As noted before, the PicoHarp 330 input channels (as well as the SYNC channel) are designed electrically identical and can be configured either for Edge Trigger or CFD mode. To begin your first experiments it is recommended to start in Edge Trigger mode as it is easier to understand and easier to set up. We assume Edge Trigger mode in the following.

For precise timing the input signals must consist of pulses with steepness and amplitude matching the specifications. The trigger edge should be set to the leading edge (rising=1, falling=0). For example, a NIM type signal is appropriate. This is a steep negative pulse (0.5 to10 ns wide, active edge falling) of typically −800 mV into 50 Ω. The PicoHarp 330 can handle up to ±2 V but large amplitudes may cause excess interference and crosstalk between the inputs. Amplitudes around 200 to 500 mV (on all inputs) are best in terms of timing accuracy and lowest histogram ripple. It may therefore be advantageous to attenuate NIM pulses by 10 to 15 dB. Lowest crosstalk is typically achieved by using signals of similar amplitude on all inputs. SMA in-line attenuators of suitable bandwidth can be used to adjust this. Note that popular TTL–SPAD–detectors (e.g., Perkin–Elmer/Excelitas SPCM–AQR) deliver positive pulses of ~3 V and must be connected through an attenuator or a pulse inverter with attenuation (PicoQuant SIA 400). **Connecting TTL signals directly will cause damage to the PicoHarp 330!**

PMTs should be connected through a preamplifier (10 to 20 dB). MCP–PMT detectors should be connected through an amplifier with slightly higher gain. All accessories are available from PicoQuant. Be sure to switch the high voltage supply of PMTs off and allow their electrodes to discharge before connecting / disconnecting them. Their high voltage charge may damage the preamplifier. Observe the allowed input signal levels including those of the pre–amplifier. Again, in a new experimental setup, to be absolutely sure, please check your detector pulses as well as the preamp output with a fast oscilloscope. Start timing is on the leading edge, so it should be

steep. Ringing and overshoot should be as small as possible. Do not over–illuminate the detector to avoid damaging it.

If the signals are appropriate, connect a detector at channel 1. Starting with only one detector makes the first steps easier. Preliminary adjustments can actually be done with uncorrelated light, e.g. daylight. To protect your detector, use strongly attenuated light, even when the detector is off. Start the PicoHarp 330 software and open the Control Panel. Look at the Control Panel section for the chosen channel. Select Edge Trigger and initially set the trigger level to half the expected pulse height and set the trigger edge (rising=1, falling=0) to match the leading edge of your detector pulses. *Tdead* is a programmable dead time for suppression of afterpulsing artefacts of some detectors. Leave it at 0 for now.

The input rate meter (next to the sync rate meter) should now show the input rate(s). Make sure that there is actually a signal coming in (some strongly attenuated light on the detector) and try to adjust the trigger level. To monitor this, watch the count rate meter. Precise tuning of the level settings is only critical for PMTs or HPDs. For such detectors an initial setting of the trigger level of -50 mV (assuming a non-inverting amplifier, otherwise +50 mV) should be approximately right to suppress electrical noise. Moving this level towards zero should result in an increased count rate due to increased detection of small pulses and pickup of noise. If you move the trigger level farther away from zero, you will see a reduced rate that will incur losses of true counts if you go too far. Ideally you should stay just slightly above the noise level. If you still see millions of counts per second at -100 mV, reduce the detector illumination. Your detector may be at risk at such high count rates. You can then also check the responsivity to illumination changes. For SPAD detectors that typically deliver pre–shaped pulses of constant amplitude the setting of the trigger level is very simple. Just set the trigger level to approximately half of the pulse amplitude.

To actually collect histograms, select a measurement range large enough (determined by the chosen resolution) and a display range to cover your sync interval (i.e. $1 / f_{sync}$) if possible. Set Offset = 0 and StopAt = 4,294,967,295. Start a measurement in oscilloscope mode with e.g., 1 second acquisition time (see the next section for instructions). Once counts are coming in, you can try to set an upper limit for the count axis so that the histogram is scaled for best viewing. Start with logarithmic count axis scaling so that weak signals can be seen.

With a periodic sync source and uncorrelated light you should obtain a more or less evenly filled histogram of a width corresponding to the sync period. When the overall input rates are very high it may happen that sync events get lost and events appear outside the sync period. It may then be necessary to increase the sync divider. Otherwise keep the divider small to avoid unnecessary jitter.

If you used uncorrelated light so far, you may now want to move on to time–correlated measurements. With a detector signal that is e.g., induced by a laser or a 'fake' electrically derived from the laser sync, you can try to obtain a histogram that should be a narrow peak, as opposed to the uniform distribution of uncorrelated light. This requires an experimental setup that delivers the signals at the two input channels with a more or less constant relative delay. In histogramming mode this delay must be such that the signal of Channel 1..N (detector) comes after the sync signal (forward start–stop mode). In addition, the delay must be chosen so that it fits in the measurement range of the histogrammer. To obtain such a timing it may be necessary to adjust the input offsets or the relative cable lengths, while also considering any optical delays.

When the rough setup is complete and a time–correlated signal peak is present in the histogram you may want to switch to a higher resolution. For optimization purposes you can then try to slightly vary the trigger levels (including sync) for best timing response. The Full Width at Half Maximum (FWHM) of the signal peak is displayed on-line as a figure of merit for instrument response. If there is no clear optimum for any level, return to the center of its stable working range.

If your detector exhibits fluctuating pulse heights, as known to occur with PMT, MCP-PMT and HPD, and provided that the pulses are negative, it may be beneficial to switch the input trigger method to CFD mode. The programmable parameters zero cross and level must then be re-adjusted for best possible FWHM. Recommended start values are a zero cross value of -20 mV and a level of about half of the pulse peak amplitude.

You can then also adjust the input offsets to place the signal properly within the time axis boundaries between two sync pulses. You may also want to adjust the axis limits to optimize the histogram display. Note that you can use the mouse wheel for quick changes of any spin control. Note also that you can double–click all the rate meters to enlarge them as separate windows. This is useful for optical alignments at your setup if the PC screen is some distance away from it.

Input Troubleshooting

Whenever there is a problem, first check your cabling, detector power supply, and sync source. To be absolutely sure, check the signals with a fast oscilloscope (50 Ω input!). **Never use TTL signals directly**. Never try to deliver the signals to multiple 50 Ω loads in parallel using simple T–pads. Use proper power splitters instead. If you have multiple detectors, try to test your setup with one detector first to keep things simple. Never forget to let detectors using high voltage discharge before connecting/disconnecting.

If you cannot get a stable sync rate reading with expected values, there may be several reasons:

- there is no proper sync signal (voltage, polarity, pulse width, frequency)

- the sync trigger level setting is inappropriate

- the sync divider setting is inappropriate

- at small rates the meter display may fluctuate between some discrete values, this is normal

Reasons for zero input channel counts may be:

- trying to manipulate the wrong input channel

- no or inappropriate signal (voltage, polarity, pulse width, frequency)

- inappropriate trigger levels

- broken cables or connectors, preamplifier or detector failure

Note that all meters are updated at the display refresh rate you have selected in the general settings dialog. In oscilloscope mode the update frequency is equal to the acquisition time. If nothing seems to happen at all, you may be in oscilloscope mode with a very long acquisition time.

Once count rates > 0 are being displayed you should then also see counts appearing in the histogramming rate meters. Make sure your measurement range is large enough. Also make sure the offset value in the Acquisition tab is set to 0. If there are histogramming counts but no histogram is building up on the screen, check the time and count axis bounds. It is best to start with a wide display range and then narrow it down. Similarly, a logarithmic count axis scaling is the safest way to see even small histograms. If your measurements stop earlier than expected, make sure that 'Stop at' is disabled, unless you have experimental reason to limit the counts.

During the set-up process you should pay attention to the warning icon that may appear at the bottom right of the main window. When the PicoHarp 330 software is running with functional hardware it continuously collects information about the input signals and the current acquisition settings. If these settings in combination with the input rates indicate possible errors, the software will activate the warning icon.

The warning icon can be clicked to display a list of current warnings together with a brief explanation of each warning (see also section 8.1). Note that the software can detect only a subset of possible error conditions. It is therefore not safe to assume "all is right" just by seeing no warning. On the other hand, if any of the warnings turns out to be an unnecessary nuisance, e.g., because your specific measurement conditions will expectedly cause it, you can disable that warning via the general settings dialog (see section 6.7).

5.2. Setting Up and Running Interactive Measurements

The primary mode of operation of the PicoHarp 330 software is interactive histogramming. This is what the main window of the software is dedicated to. The user can set up measurement parameters, start measurements and immediately see histogram data on the screen. In further sections, e.g., on TTTR mode, you will learn about other modes of operation with less user interaction that will collect data straight to disk without immediate visualization. Here, we focus on the interactive histogramming mode of operation.

To set up measurement parameters use the PicoHarp 330 control panel. The control panel can be opened by clicking the control panel button on the toolbar or by pressing <Alt>+C.

In the control panel section 'Acquisition' you can set the resolution (time per bin), the offset, the measurement time, and the block of memory to use for this measurement. To begin, use a measurement time of 1 second and an offset of 0. There are always 65,536 time bins per histogram. Histograms can be recorded and stored in 512 memory blocks. Out of these up to 16 curves can be displayed and one 'active' block can be used for a measurement. You can designate the active memory block you wish to use for the next measurement by selecting the block number in the control panel. Never forget to select a new block when collecting new data and old data is intended to be preserved.
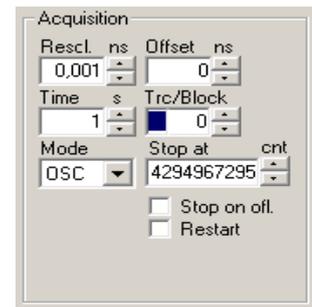
Use the trace mapping dialog to select up to 16 curves for display. Make sure the memory block you measure into is mapped to a display trace that is switched on, so that you can see the trace. You can reach the trace mapping dialog from the toolbar or by clicking on the trace color indicator next to the block selector.

The color indicator shows the trace color that the chosen block is currently mapped to. If it is a solid square, the curve is mapped and shown. If it is mapped but not shown, the indicator shows a small striped square. If the curve is not even mapped for display the indicator remains white.
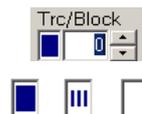
There are two basic histogramming modes for interactive measurements: Oscilloscope and Integration mode. Oscilloscope mode repeatedly collects histograms with a fixed measurement time and displays them on the screen. This lets you see fast changes in the histogram, e.g., for optical setup and adjustments. Usually this only makes sense with relatively strong signals and short acquisition times. Integration mode is usually operated with longer acquisition times. In this case the histogram continues to grow over a longer time and the display is updated at regular intervals, so that the accumulation process can be observed.

To start a measurement with the current control panel settings, use the start button (GO) on the toolbar or press <Alt>+G.

To stop a measurement use the stop button on the toolbar or press <Alt>+S.
Note that a measurement may automatically stop and / or restart, dependent on the current settings of '*stop at*' and '*restart*' in the control panel.

To actually run a meaningful measurement you will at first need to set up the input channels, most importantly with appropriate voltage levels as outlined in the previous section. Also allow a warm–up period of about 20 minutes (depending on lab temperature) before using the PicoHarp 330 for final measurements. You can use this time for set–up checks and preliminary measurements.

The rate meters (bottom of main window) permit visual control of the data acquisition. The meters are often too small to view from further away, e.g., when adjusting the optical setup. You can in this case simply double–click the rate meter of interest. This opens a larger window that you can then re-size and re-position on the screen as you like.

Once you have established standard settings for your experimental setup you may want to save them to a file. The control panel settings can then be recalled at any time by loading that file. The software stores all settings together with the histogram data of those curves in memory that have been filled by a measurement. In addition to this, all settings are stored in the Windows registry, so that at program startup you find the control panel as it was when you last closed the PicoHarp 330 software.

5.3.   Time Tagged Mode Measurements

Time–Tagged Time–Resolved (TTTR) mode allows the recording of individual count events directly to hard disk without immediately forming histograms.

In classic TTTR mode, in addition to the start–stop timing with picosecond resolution, the timing of the events with respect to the beginning of the whole measurement is recorded in the event records. This is particularly interesting when the dynamics in a fluorescence process are to be investigated. The availability of the time–tags permits photon burst identification, which is of great value e.g., for Single Molecule Detection (SMD) in a liquid flow. Other typical applications are Fluorescence Correlation Spectroscopy (FCS) and Burst Integrated Fluorescence Lifetime (BIFL) measurements. Together with an appropriate scan controller, TTTR mode is also suitable for ultra fast Fluorescence Lifetime Imaging (e.g. rapidFLIM). Another great application area of time tagging is quantum optics with coincidence counting and correlation.

The PicoHarp 330 supports two different Time–Tagging modes, T2 and T3 mode, which will be explained further below. When referring to both modes together we use the general term TTTR.

5.3.1.   System Requirements

In cases where the Time–Tagging modes are to be used with high continuous count rates (say > 20 Mcps) the PC system must meet some special performance criteria. The reason for this is the relatively large amount of data being generated in TTTR mode. In order to prevent a buffer overrun in the recording, the data must be transferred to the computer in real–time. This requires a modern PC with a fast I/O subsystem. A recent, at least quad core CPU running at 2 GHz or more is required. For the best possible performance in TTTR mode a modern solid state disk with high throughput is recommended. If it is intended to make use of the full TTTR throughput of a PicoHarp 330 (up to 90 Mcps via USB 3) then the hard disk must be able to handle sustained write rates of 360 MBytes/s. This can be achieved with RAID arrays or modern solid state disks. Network storage is usually too slow.

5.3.2.   T2 Mode

In T2 mode all timing inputs of the PicoHarp 330 including the sync input are functionally identical. There is no dedication of the sync input channel to a sync signal from a laser. It may be left unconnected or can be used for an additional detector signal. In this case the sync divider must be set to "None". Usually the regular inputs CH1..N are used to connect photon detectors. The events from all channels are recorded independently and treated equally. In each case an event record is generated that contains information about the channel it came from along with the arrival time of the event with respect to the overall measurement start. The timing is recorded with the highest resolution the hardware supports (1 ps). Each T2 mode event record consists of 32 bits. There are 6 bits for the channel number and 25 bits for the time–tag. If the time tag overflows, a special overflow record is inserted in the data stream, so that upon processing of the data stream a theoretically infinite time span can be recovered at full resolution. Dead times exist only within each channel but not across the channels. Therefore, cross correlations can be calculated down to zero lag time. This allows for powerful applications such as coincidence correlation and FCS with lag times from picoseconds to hours. Autocorrelations can also be calculated at the full resolution but of course only starting from lag times larger than the dead time.

The 32-bit event records need to be continuously streamed to the host PC, ideally without losses. Given the dead time of the TDCs of Td=680 ps it is theoretically possible that event records are generated at a maximum rate of 1/Td = 1,47 GHz. The total rate increases even more when multiple channels are used. It is obvious that such data rates cannot be transferred over USB continuously without losses. On the other hand, given typical photon statistics one can (and should) distinguish between peak rates and average rates. The latter are typically much smaller than the peak rates during photon bursts. The system is therefore designed to use FIFO (First In First Out) buffers that can temporarily hold a certain number of events in a burst so that the bus transfer must only deal with moderate average rates. In order to allow a high input burst rate the PicoHarp 330 uses in each channel a fast front-end FIFO buffer that can handle bursts of up to 1000 events at the maximum rate of 1/Td = 1,47 GHz. This is followed by a sorter and a large but slower FIFO buffer for all channels, capable of holding up to 268,435,456 event records. This large secondary FIFO buffer ensures that no data is lost due to inevitable task switches and interruptions of the USB transfers on the host PC side. Even if the average read rate of the host PC is limited, bursts with much higher rate can be recorded for some time. Only if the average count rate exceeds the readout speed of the PC over a long period of time, a FIFO buffer overrun can occur. In case of a FIFO buffer overrun the measurement must be aborted because data integrity cannot be maintained. However, on a modern and well configured PC a sustained average count rate of up to 90 Mcps is possible. This total transfer rate must be shared by the inputs used. For all practically relevant photon detection applications the ef-

fective rate per channel is more than sufficient. If there are very intense bursts at the input of the front-end FIFO it may happen that events are lost. Similarly the sorter may be overloaded so that events must be dropped. This is indicated to the software by means of a hardware flag so that the user can be informed of such losses. The user must then decide if the losses can be tolerated for the given experiment (see also section 8.1 on warnings).

For maximum throughput, T2 mode data streams are normally written directly to disk, without preview other than count rate and progress display. However, it is also possible to analyze incoming data "on the fly". The Pico-Harp 330 software provides a real-time FCS correlator for preview during a T2 mode measurement (see section 5.3.7). Other types of real-time processing must be implemented by custom software. The PicoHarp 330 software installation provides demo programs to show how T2 mode files can be read by custom software (see the folder `filedemo` under the chosen software installation folder). The implementation of custom measurement programs requires the PicoHarp 330 programming library, which is provided as a separate item in the PicoHarp 330 software package. A relatively advanced high-level API package for Python called "snAPI" is also available (separate download from GitHub). It readily provides many real-time analysis methods such as intensity and coincidence time traces, FCS and $g^{(2)}$ correlation. Alternatives for advanced T2 data collection and analysis are the SymPhoTime and QuCoa software suites offered by PicoQuant. SymPhoTime is focused on typical life science applications while QuCoa is oriented towards typical quantum optics applications.

5.3.3.   T3 Mode

In T3 mode the sync input is dedicated to a periodic sync signal, typically from a laser. As far as the experimental setup is concerned, this is similar to classic TCSPC histogramming. The main objective is to allow for high sync rates which could not be handled in T2 mode. Accommodating the high sync rates in T3 mode is achieved as follows: First, the sync divider is employed (as in histogramming mode). This reduces the sync rate so that the front end processing throughput is no longer a problem. The remaining problem is now that even with the divider, the sync event rate may still be too high for collecting all individual sync events like in ordinary T2 mode. Considering that sync events are not of primary interest, the solution is to record them only if they arrive in the context of a photon event on any of the input channels. The event record is then composed of two timing figures: 1) the start–stop timing difference between the photon event and the last sync event, and 2) the arrival time of the event pair on the overall experiment time scale (the time tag). The latter is obtained by simply counting sync pulses. From the T3 mode event records it is therefore possible to precisely determine which sync period a photon event belongs to. Since the sync period is also known precisely, this furthermore allows reconstructing the arrival time of the photon with respect to the overall experiment time.

Each T3 mode event record consists of 32 bits. There are 6 bits for the channel number, 15 bits for the start–stop time (dtime) and 10 bits for the sync counter. If the counter overflows, a special overflow record is inserted in the data stream, so that upon processing of the data stream a theoretically infinite time span can be recovered. The 15 bits for the start–stop time difference cover a time span of 32,768×R where R is the chosen resolution. At the best possible resolution of 1 ps this results in a span of about 32 ns. If the time difference between a photon and the last sync event is larger, the photon event cannot be recorded. This is the same as in histogramming mode, where the number of bins is larger but also finite. However, by choosing a suitable sync rate and a compatible resolution R, it should be possible to reasonably accommodate all relevant experiment scenarios. R can be chosen in a wide range, starting with the device's base resolution and then continuing by repeated doubling of the time bin width.

Dead time in T3 mode is the same as in the other modes. Within each photon channel, autocorrelations can be calculated meaningfully only starting from lag times larger than the dead time. Across channels dead time does not affect the correlation so that meaningful results can be obtained at the chosen resolution, all the way down to zero lag time. This requires custom software.

The 32 bit event records are queued and forwarded to the host PC in the same staggered FIFO architecture as described in the section on T2 mode above. Accordingly, a sustained average count rate of up to 90 Mcps is possible in T3 mode too, while now the sync events do not consume any transfer bandwidth.

For maximum throughput, T3 mode data streams are normally written directly to disk. However, it is also possible to analyze incoming data "on the fly". One such analysis method is the on–line correlation implemented in the PicoHarp 330 software (see section 5.3.7). Other specialized analysis methods must be implemented via custom software. The PicoHarp 330 software installation provides demo programs showing how T3 mode files can be read (see the folder `filedemo` under the chosen software installation folder). The implementation of custom measurement programs requires the PicoHarp 330 programming library, which is provided as a separate item in the PicoHarp 330 software package. An alternative for advanced T3 mode data collection and analysis is the snAPI library for Python or the SymPhoTime software suite offered by PicoQuant.

5.3.4.   Running a basic TTTR Mode Measurement

A TTTR mode measurement (T2 or T3 mode) will typically be started after all control panel settings have been tested in normal interactive histogramming mode (oscilloscope or integration). The acquisition time (measurement time) and the file for saving the data are the only parameters that can be set separately.

A typical approach to set up a TTTR mode measurement would be by first starting oscilloscope mode with an acquisition time of e.g., 1 second. Then all control panel settings should be optimized to reliably obtain the expected data.

Once all settings are satisfactory, click the "*TTTR Mode*" button 🖲️ on the toolbar. This will bring up the "*TTTR Mode*" dialog.
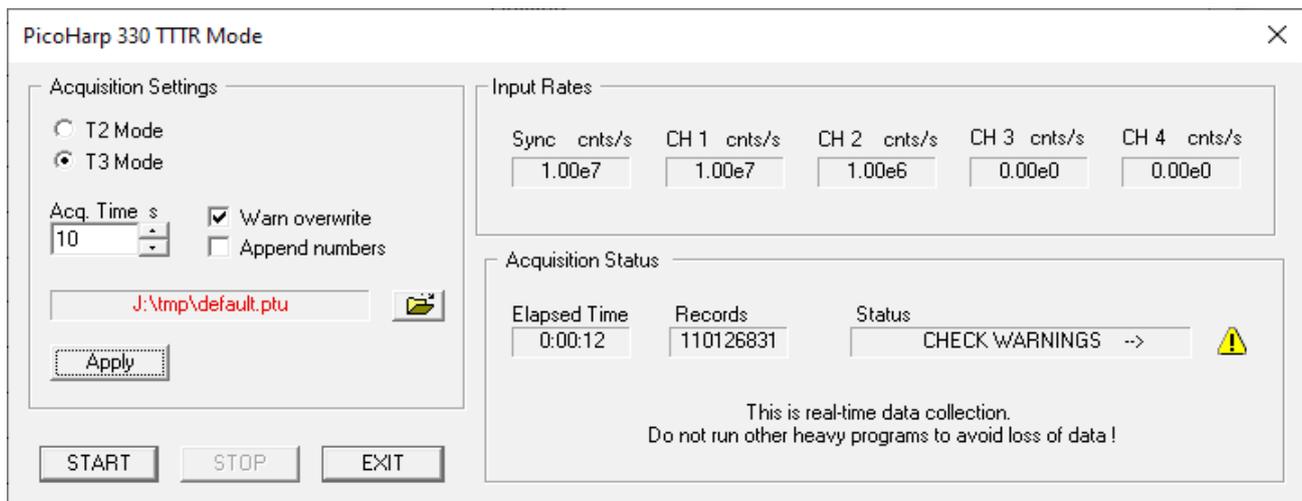
The dialog section *Acquisition Settings* allows selecting the measurement mode (T2 / T3), overall acquisition time, and file name. Note that switching between T2 and T3 mode takes some time because the hardware must be reconfigured. Normally such a switching should not occur often because the two modes usually require a different experimental setup.

The section *Acquisition Settings* also has two tick boxes for the handling of existing files. You can turn on a warning and / or automatically have numbers appended to the file names, so that you can conveniently perform series of measurements. The file name is shown in red if the file already exists. The button with the file icon will open a standard Windows file dialog. You can select an existing file or choose a new name. The PicoHarp 330 TTTR mode files have the extension ".ptu". For maximum count rate throughput you should choose a file destination on a fast local hard disk as outlined above. Network drives are often too slow.

The dialog section *Input Rates* shows the present count rates of all channels. The dialog section *Acquisition Status* shows elapsed time, the number of collected records and a status line showing what is currently happening.

Next to the status line you may see a warning icon. When the TTTR mode dialog is running it continuously collects information about the input signals and the current acquisition settings. If these settings in conjunction with the input rates indicate possible errors, the software will display the warning icon. The warning icon can be clicked to show a list of current warnings together with a brief explanation of each warning (see also section 8.1). In order to fix wrong settings you may have to close the TTTR mode dialog and return to the control panel.

Down on the left there are buttons for Start, Stop and Exit. Start and Stop control the actual TTTR measurement run. Exit is for leaving the TTTR mode dialog.

The TTTR mode measurement will start as soon as you click the Start button. You will then be able to watch the progress of your measurement in the status boxes showing the elapsed measurement time and the number of events that have been recorded. Note that this includes overflow and marker records. Therefore, the number of records will be somewhat larger than the true photon counts. The overflow and marker records will later be filtered out by the data analysis / processing software.

A measurement can be stopped at any time by clicking the Stop button. The data recorded up to this point will be stored in the file. When the measurement is completed, the Stop button will be grayed out (disabled). Use the Exit button to return to the normal interactive mode. Again, this will take some time for hardware reconfiguration.

As outlined above, TTTR data collection at high rates is a demanding real-time streaming process. The hardware and software must ensure not to lose any data. In order to implement this efficiently the PicoHarp 330 software employs multiple threads (concurrent CPU processes). A first thread continuously reads the PicoHarp 330's hardware FIFO and puts the retrieved data in a software queue. A second thread concurrently reads this software queue and writes the data to disk. If real-time correlation is being performed then this is done in further separate threads. User interface and interaction are handled in yet another thread. Multi-core CPUs are particularly useful here as they can run the threads in parallel rather than switching between them. There are two typical error scenarios that you may encounter in this process. The first is a situation where the first thread does not empty the hardware FIFO quickly enough and the FIFO runs full. The software then reports the error FIFO_OVERRUN. In order to avoid this you may have to reduce the input data rate. An option for this is hardware filtering (See section 5.3.8). Another error situation may result when the second thread cannot write to disk quickly enough and consequently the software queue runs full. The software then reports the error STORAGE_QUEUE_OVERRUN. In order to avoid this you may want to check the write speed of your hard disk and see how it can be improved.

5.3.5. External Markers

Often it is desirable to synchronize TCSPC measurements with other information or processes of complex measurement tasks. In order to perform e.g., Fluorescence Lifetime Imaging, the spatial origin of the photons must be recorded as well as their timing. For this purpose a mechanism is needed to assign external synchronization information to the TCSPC data. In the special case of Fluorescence Lifetime Imaging (FLIM), conventional systems used on–board memory and switch to new blocks of memory upon arrival of e.g., a pixel clock pulse. Accommodating the large amount of data generated by the 3–dimensional matrix of pixel co–ordinates and lifetime histogram bins is a serious challenge. Even with modern memory chips, this approach is limited in image size and / or number. In addition, it implies loss of information about the individual photon arrival times. To solve the problem in a much more elegant manner, the TTTR data stream generated by the PicoHarp 330 can contain markers for synchronization information derived from an imaging device, e.g., a scan controller. For this purpose the control port of the PicoHarp 330 provides four TTL inputs for synchronization signals M1..M4 (see section 8.3.2 for the connector specification).

The figure below illustrates how the external marker signals are recorded in the data stream.
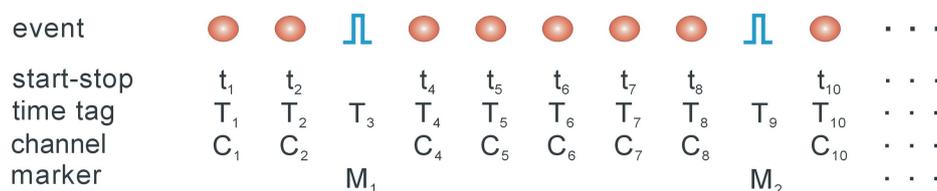
Bullets denote a photon, blue pulses denote a marker signal. The external markers are treated almost as if they were regular photon event records. A special channel code allows to distinguish true photon records from marker records. Software reading the TTTR file can thereby filter out the markers e.g., for line and frame clock in imaging applications. This makes it possible to reconstruct the 2D image from the stream of TTTR records during data analysis, even in real-time. The stream data is nearly free of redundancy and can therefore be transferred efficiently. The image size is unlimited both in XY and in count depth. Since there are four such synchronization signals, all imaging applications including 3D can be implemented and optionally even other experiment control signals can be recorded. This marker scheme is a very special feature of PicoQuant's TCSPC electronics. It may be worth noting that inventing this technology enabled PicoQuant to develop the seminal MicroTime 200 Fluorescence Lifetime Microscope.

Four TTL compatible inputs accept the synchronization signals to be recorded as markers. The active edges of these signals can be chosen in the general settings dialog (available through the Toolbar). Both high and low state must be at least 50 ns long. The period may therefore (in principle) be as short as 100 ns but data bus throughput constraints will apply. Each marker creates an additional TTTR record, so that one must ensure not to swamp the data stream with too many marker records. In case of data pipeline congestion markers take precedence over photon records, so that excessive marker traffic can suppress photon records. In fast imaging applications it is therefore recommended not to use a pixel clock but a line clock only. Since each photon has a time tag, it is usually not necessary to use an additional pixel clock. Instead, virtual pixels can be defined by

subdividing the scan lines in time. The accuracy of marker timing is on the order of 50 ns. Note, however, that in T3 mode it can never be resolved better than one sync period.

A programmable marker hold-off time can be used to suppress glitches on the marker signals that some poorly designed scan hardware or cable reflections might create. The idea is that when a marker signal was detected the next (spurious) marker will be suppressed if it occurs within the hold-off time after the first detection. The hold-off time can be chosen in the software settings dialog available through the Toolbar.

5.3.6. Using TTTR Mode Data Files

For diagnostic purposes you may reload a T3 mode file into the PicoHarp 330 software. The limitation is that you will only be able to form a histogram over the start-stop times in your T3 mode data. The time-tag information will not be used here. The PicoHarp 330 software will recognize that you are loading a T3 mode file and how many records are contained in it. It will then prompt you for a range to use for histogramming. The histogram will go to blocks 0..N-1 of the histogram memory where N is the number of channels that were used in the T3 mode measurement. A TTTR mode file also contains all control panel settings that were active at the time of the measurement run. After loading a TTTR mode file, you will find the document title reading "Histogram from...". If you choose to save such data you will have to give it a new file name (`*.phu`). This is because now a histogram has been formed, and saving it with the same file name would destroy the original TTTR mode file. However, you may save the previously formed histogram as if it were obtained in normal interactive mode, to a standard PicoHarp 330 histogram data file (*.phu).

Reloading T3 mode files serves as a quick diagnostic tool only. For T2 mode files such a feature is currently not available. Further processing or analysis of TTTR mode data must therefore be performed through external data analysis software. Such software is available from PicoQuant for a wide range of analysis tasks (under the product names SymPhoTime and QuCoa). Further specialized analysis can be performed by dedicated custom software. If you wish to save the cost for the commercial TTTR analysis software or if you require special analysis algorithms you may want to program your own analysis software. The implementation of custom measurement programs requires the PicoHarp 330 programming library, which is provided as a separate item in the PicoHarp 330 software package. An advanced high-level API package for Python called "snAPI" is also available (separate download from GitHub). It readily provides many real-time analysis methods such as intensity and coincidence time traces, FCS and $g^{(2)}$ correlation. The same methods can also be applied to recorded files.

For development of your own highly custom data analysis programs, you may also want to refer to the demo code for loading `.phu` and `.ptu` files. Demo source code is included in your PicoHarp 330 installation location under the subfolder `filedemo`. Also see section 8.2 for the file format specifications. The paragraph below gives only an outline.

The first part of a TTTR mode file is a header with the basic setup information, similar to that of the other modes. What follows after the header is a sequence of 32 bit TTTR records. The TTTR records in the file consist of different pieces of information in groups of bits. These pieces of information must be extracted by a bit masking / shifting operation. Their specific layout is different for T2 and T3 mode. In both cases, in addition to extracting the bit fields, the software must step through the whole file and interpret the overflow records and correct the overall time axis accordingly. Details on how this can be done should be looked up in the demo programs that are installed as part of the software distribution.

5.3.7. TTTR Mode Measurements with Real–Time Correlation

An important application of the PicoHarp 330 and its TTTR mode is Fluorescence Correlation Spectroscopy (FCS). While TTTR data collection and off–line software correlation has been used for quite some time, PCs have only recently become fast enough to calculate the correlation function "on the fly" while the data is being collected. This capability can be very useful in setting up and monitoring FCS experiments. The PicoHarp 330 software provides such a real–time software correlator for T2 and T3 mode. Note that this is not a correlator for $g^{(2)}$ correlations common in quantum optics. If you need this type of correlator please consider PicoQuant's QuCoa software or the new snAPI package for Python.

Although it collects the same type of data as regular TTTR mode, the real–time correlator has its own button on the Toolbar.

The next figure shows the TTTR mode correlator dialog.

The lower section of the dialog provides the same file handling and progress indicator elements as in regular TTTR mode. Please refer to section 5.3 for details. The upper section of the dialog contains the correlator display and various control elements. The dialog section *Cor(Ax,Bx)* allows to select individual input channels for the correlation. A and B are the virtual input channels to be correlated. All physical input channels that are ticked

in columns A and B, respectively, will be aggregated and used in the corresponding correlation channel. If your selection is empty, the heading of the section *Cor(Ax,Bx)* will turn red to indicate the problem. If you proceed anyway you will get error messages upon start.

The correlator always calculates the autocorrelations of the two virtual channels (AA and BB) as well as the cross-correlation (AB). There are three tick boxes that select which of these are shown.

The dialog section *Accumul.* allows selecting an update time for the correlation display. Furthermore it allows to select between repetitive updates (*Osc.*) and cumulative collection (*Int.*). If the latter is chosen one can manually reset the correlator by clicking the *Reset* button. Note that all these settings affect only the real-time correlator. The raw TTTR mode data will be collected continuously and completely independent from the correlator settings.

The *Save* button under the *Accumul.* section allows to save the correlator results as they were last shown. The saved result is an ASCII file with some header information. The format is self-explanatory.
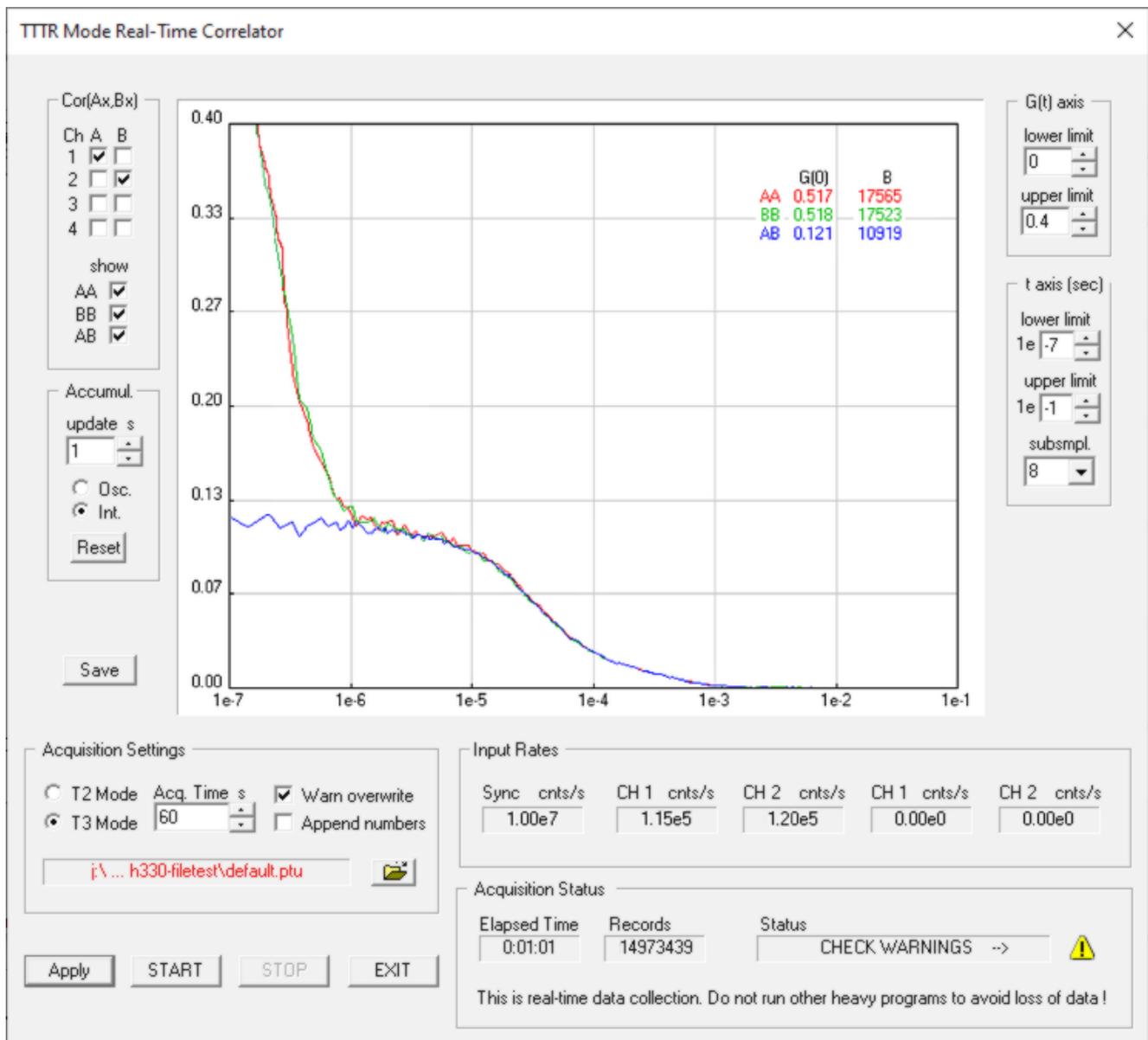
On the right hand side there are controls for the axis ranges of the correlation plot. Note that they affect only the display. Collected data is always complete, independent from the axis settings. In contrast, the selection box *subsmpl.* has an effect on the correlator results. It determines how many tau points are calculated. The correlator works in a logarithmic multiple tau scheme and *subsmpl.* specifies the number of linear subsamples in each

log stage. A higher number of subsamples increases the resolution of the correlation curve. Calculating more points is more time consuming and therefore may lead to lower count rate limits that can be handled. In this case you may get FIFO overruns. The default of *subsmpl.*=8 is a reasonable trade–off favoring speed over resolution. On faster computers the setting of *subsmpl.*=16 is a most likely feasible choice for better resolution.

Note that the starting point and spacing of the tau sampling points also depends on the time tag resolution. In T3 mode this corresponds to the sync period. In T2 mode the native resolution of the board is binned down to a time tag resolution of 25 ns in order to make the data manageable for the real-time correlation algorithm.

The correlation curve display can be shown with a grid (see general settings dialog, accessible from the toolbar of the main window). It also shows two useful figures obtained from the collected data (top right of curve window). The first figure is an approximation of G(0). In classic FCS experiments this corresponds to the inverse of the number of particles in the focal volume. It is continuously updated together with the curve display, which can be useful for system adjustments, notably when using the repetitive accumulation mode (Osc.). Note that the approximation is a simple averaging over the first ten tau points. The figure B is an indicator for molecular brightness. It is also updated continuously. Note that it is also only an approximation obtained by multiplying the G(0) approximation with the average count rate on both virtual correlation channels A and B. Depending on the chosen channels in A and B this may lead to figures that do not truly reflect molecular brightnesses. However, they should be useful as an adjustment aid in any case.

All other aspects of TTTR data collection with correlator preview are the same as in plain TTTR mode as described in the previous subsections in chapter 5.3.

Note that the real-time correlator is primarily a tool for preview during measurement setup and optimization. It does not allow to correlate data from TTTR files. If you need the correlation over the entire stream of collected data you need to make sure that the measurement runs in Integrating mode and that you do not press reset during the acquisition. If you need the correlation results you should save them here. If you need to perform further or more thorough FCS analysis on the collected data you can use the SymPhoTime 64 software from PicoQuant or, if you wish to build your own tools, use the elegant new snAPI library for Python.

5.3.8.   TTTR Mode Measurements with Event Filtering

Introduction and Application Context

In many quantum optical application scenarios, information is encoded in the correlation of two or more photons. Traditionally, coincidence measurements are used to evaluate these correlations experimentally. These range from relatively simple experiments like bipartite entanglement measurements to more complex setups as used in sampling or universal quantum computation experiments. The following figure sketches a stream of correlated photon pairs across two detection channels, as for example generated by spontaneous parametric down-conversion (SPDC) pumped with a pulsed laser of repetition rate f_{rep}.
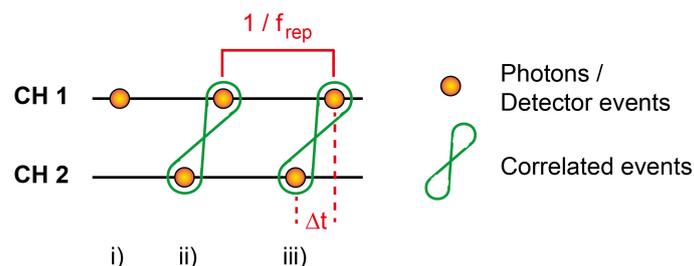
Due to optical and electrical delays the correlated photons may arrive at the detectors at different times. This creates a time offset Δt between the event in detection channel CH1 and the event in detection channel CH2 that belong to one correlated photon pair (green loops at time positions ii and iii).

Historically, early detection hardware like coincidence logic analyzers, required that all signals needed to be temporally well synchronized. This corresponds to negligible Δt in the sketch above and the correlated events in CH1 and CH2 happen simultaneously – they coincide temporally. A coincidence count is then just the logical combination of these two detection events. In coincidence logic analyzers this combination is carried out in hardware. In order to allow for small tolerances it is good practice to allow for a range of temporal offsets Δt that a coincidence analyzer still counts as coincidences – this range is often called coincidence time window. Using modern time tagging TCSPC systems, there is no need for an a priori synchronization of the detector channels.

In such systems, the time offset between the channels can be arbitrarily shifted in hardware and/or postprocessing. Consider a time offset Δt even larger than the one shown in the sketch above. Here, a photon from an earlier pair in CH1 (i – no green loop) would falsely be aligned with a photon from a later pair in CH2 (ii). Counting this event as a coincidence would not allow to reveal the true correlations. This example illustrates the advantage of modern TCSPC systems: when using a hardware coincidence logic analyzer the whole dataset of the measurement would be lost in the case of a wrong delay alignment. With modern time tagging TCSPC systems the correct correlations can be easily recovered by adjusting Δt during data analysis or thereafter.

However, coincidence logic analyzers also have their unique advantage. In the sketched quantum optics application scenario, the correlations of interest only exist between pairs or higher numbers of photons. Often lower-order events, e.g., single events in the case of two-fold coincidences contain little information for these experiments. Coincidence logic analyzers already discard these lower-order events and hence reduce the recorded data size, often drastically. This does not only make data analysis less cumbersome and file size smaller, but also allows for higher data rates for the signals of interest, as the bus interface to the host PC is not burdened with unnecessary data. Hence, data can be processed and displayed on a host PC in real time.

The event filter implemented in gateware combines the advantages of TCPSC devices with the advantages of traditional coincidence logic analyzers. In order to make the filter beneficial for a wide range of quantum optical application scenarios it is designed to operate on all input channels in use and hence the full set of their possible combinations. Traditional coincidence logic analyzers often predefine logical combinations between channels (coincidence patterns) to be recorded, while all other events are discarded. This may limit applications as the set of predefined patterns is usually hardware limited. The event filter strives to resolve this by defining discardable orders of events (e.g. single events) and keeping what may be of interest. This reduces the data load sufficiently for lossless bus transfer and reasonable file size. Computing logical combinations between channels in the remaining data can then be easily done on the host PC at virtually unlimited complexity.

In the simplest setting, the filter discards all events that do not have partners (in any channel) within the coincidence window of programmable width. This already permits a significant data reduction without loss of generality for any subsequent software analysis. For further reduction the order of filtering (the number of partners) can be set to a higher number by way of a parameter "match count", so that only 2-fold, 3-fold,…, up to 6-fold coincidences are recorded.

In contrast to many traditional coincidence logic analyzers and most of the current time tagging systems, all raw events of the contributing channels passing the filter are fully recorded with their time tag. Their complete information content is therefore available for refined analysis in subsequent software processing. Setting the time window (set by the filter parameter 'range') can therefore not only act as an immediate coincidence time window, but initially can also be set broader for a first mild data reduction. As all events are precisely time tagged, a narrower coincidence time window (or multiple such windows) can always be applied in post-processing. The "range" setting up to a certain degree tunes how close the system will behave to a plain TCSPC or time tagger without event filter and, in the opposite, a generalized coincidence logic analyzer. Note, that for the hardware event filter to work properly the channels need to be temporally aligned. This is easily achieved by means of the programmable offset in each of the input channels.

In addition to the filter functionality outlined so far, all event timing channels can optionally be marked as 'pass channels' that will be passed through the filter unconditionally. This can be very useful to provide reference channels, e.g., if the channel counts contain information to track drift of the setup in the same measurement.

As a further means of flexibility and for test purposes the filter provides an optional 'inverted' setting. In that case the filter discards all events that do have partners according to "match count".

## Event Filter Usage Walk-Through

In the following we discuss a simple experiment showing how the filter works and how it can be programmed. The experimental setup we will be using throughout this section is shown in the next figure. For the sake of clarity and repeatability we use signal generators of fixed frequency rather than the mostly non-deterministic signals usually found in quantum optics.
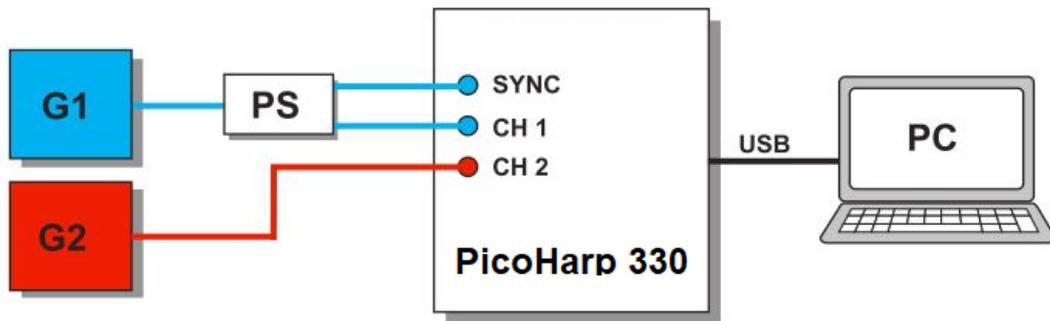
Generator G1 provides pulses at a frequency of 10 MHz. By means of a 50 Ohms power splitter (aka reflection free T-pad) identical pulse trains are fed to the PicoHarp 330's sync input and photon timing input channel 1. Generator G2 provides an independent periodic pulse train of 1 MHz that is fed to the PicoHarp 330's photon timing input channel 2. The generator frequencies and the choice of input channels are somewhat arbitrary and we only pin them down here for consistency with instrument settings and experimental results shown further on.

While event filtering is a feature available solely in TTTR mode, it is helpful to first consider what kind of results they would generate in histogramming mode. Indeed the experimental setup was designed to permit this and for didactic purposes you are encouraged to try it out. Given that the signals from G1 are identical on sync and input channel 1, the time difference between them is constant, except for a small stochastic measurement error of a few ps. Therefore, one would expect a histogram looking like a sharp spike, in fact a gaussian distribution with a width corresponding to the instrument's timing uncertainty of a few ps, provided we run it at the best possible resolution. In order to cover the full Sync period we actually need to run it at lower resolution but this does not matter here. With identical cable lengths after the splitter we should expect the peak to sit at time zero of the histogram axis. Since the histogram results from positive timing differences between input channel and sync, we would only be able to see the positive half of the peak. In order to see the full peak it is useful to introduce a delay in the input channel. This can be done physically by means of cable, but one great feature of PicoQuant's TCSPC systems is that the same can be achieved by means of a programmable offset, so we use the latter and set it to 30 ns. The signal from G2 is temporally independent from G1 so that the corresponding time difference is taking all possible values within the sync period, i.e. the 100 ns period of G1. Over some sufficient measurement time we would therefore obtain a histogram that is evenly filled, with only some statistical fluctuations according to counting statistics. The figure following below shows the actual measurement results.
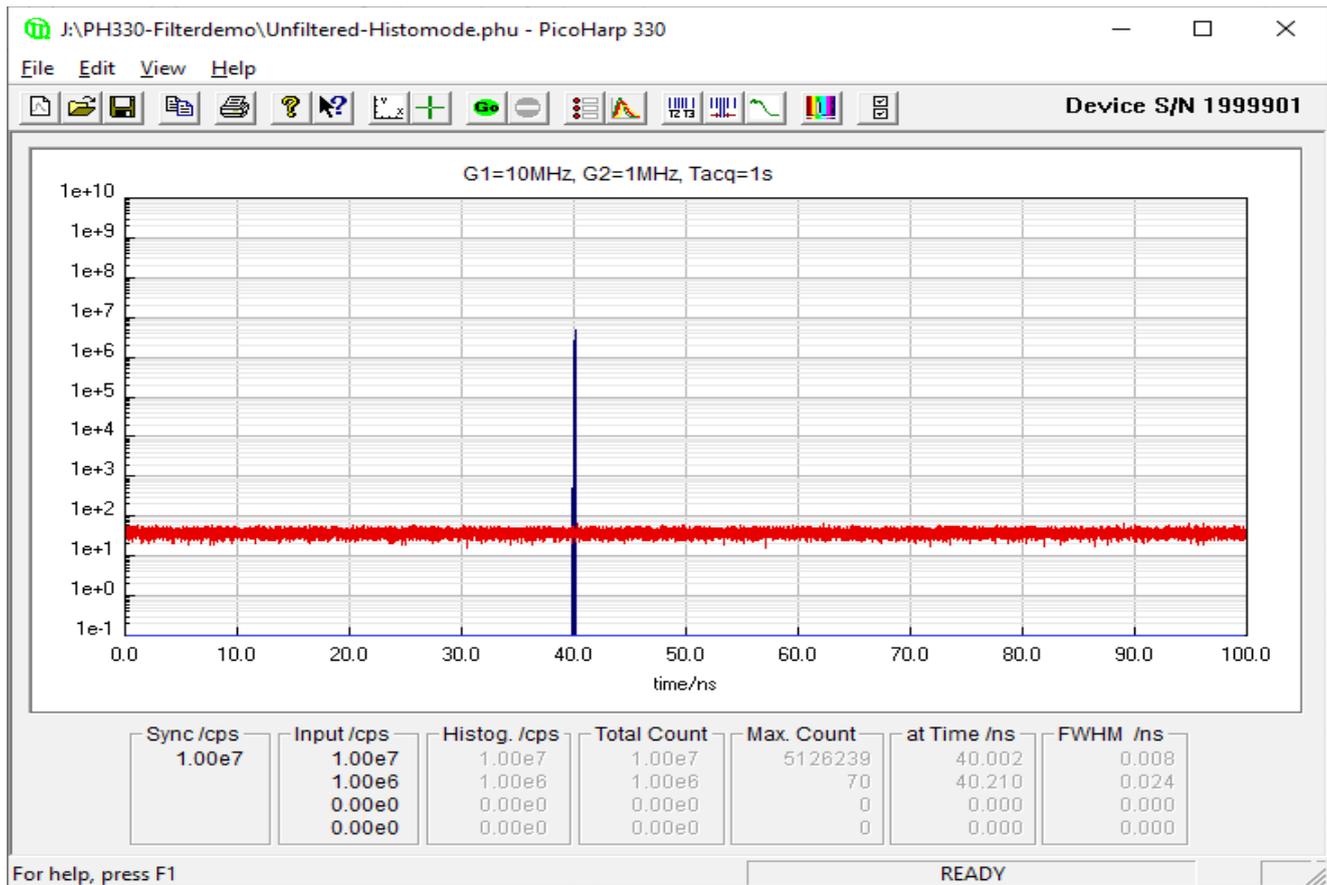
Now we take one step further and perform the same experiment in TTTR mode, first without filtering and then with filtering. We use T3 mode because it is functionally related to histogramming mode, in that it also assumes a typically periodic sync signal and, more important here, the PicoHarp 330 software permits loading T3 mode files directly into the histogram display. We could do the experiment by way of a regular TTTR mode measurement (See section 5.3.4) but for didactic purposes we shall do it through the dialog for Filtered TTTR mode.

The Filtered TTTR Mode dialog can be reached through the corresponding toolbar button shown here on the right. The dialog is conceptually similar to that of regular TTTR mode. Upon clicking on the button you will notice a moment of delay where the hardware is re-configured for the new measurement mode. When this is completed the dialog will appear similar to the figure following below.

Most prominent on the top there is a pair of radio buttons for T2 versus T3 mode. It is important to make this se‑lection first, as it will determine the permitted filter settings for the sync channel.

The next group box titled 'Input rates' is self explanatory and identical to what you find in the regular TTTR mode dialog. It gets updated every 500 ms

Below in the group box 'Event Filter' there is a matrix of tick boxes for the selection of whether and how individ‑ual channels are going to be used in the filter. For each channel there is one tick box 'use' to indicate if the channel is going to participate in the data collection and in the filtering scheme. In addition, for each channel there is another tick box 'pass' to indicate if the channel is to be passed through the filter unconditionally, whether it is marked as 'use' or not. The events on a channel that is marked neither as 'use' nor as 'pass' will not pass the filter, provided the filter is actually enabled, as marked by the tick box 'Enable filter' further right.

The other controls 'Range', 'Match Count', and 'Inverted logic' determine how the filter will act. 'Range' deter‑mines the time window the filter is acting on. It can be entered in whole nanoseconds or as floating point num‑bers. If you enter fractions smaller than the device's resolution the effective range will be rounded down. The parameter 'Match Count' specifies how many other events must fall into the chosen time window for the filter condition to act on the event at hand. The tick box 'Inverted logic' inverts the filter action, as you will see more clearly further down in this section. For now, let it be not inverted. Then, for instance, if 'Match count' is 1 we will obtain a simple 'singles filter'. This is the most straight forward and most useful filter in typical quantum optics experiments. It will suppress all events that do not have at least one coincident event within the chosen time range, be this in the same or any other channel marked as 'use'.

Also within the group box 'Event Filter' there is a set of buttons for loading and saving the filter settings. This is a convenience feature for quick retrieval of standard settings you may wish to keep. The filter settings are also memorized in the Windows registry as well as in each data file.

Further down you find a group box 'Filter Test'. This provides a useful feature for testing a filter configuration, and in particular checking as to whether the resulting data rate may or may not be sustainable over USB. Recall that the PicoHarp 330 has an extremely high input bandwidth: each channel can in principle capture events at up to 1.47 GHz. Despite a sophisticated FIFO buffering scheme (see section 5.3.2) combined over the instrument's channels this high rate cannot be sustained for long in internal processing and USB transfer. The filter test will permit to try this out without doing a real measurement and running into FIFO overruns. Because the test performs a quasi-real measurement, only without forwarding data from the filter to the FIFO, it cannot be performed when a real measurement is running and vice-versa. The corresponding buttons will be mutually disabled and greyed out to prevent this.

Using the setup with the two generators as sketched above we can now perform a filter test by clicking the Run button and observe the Input and Output rates:
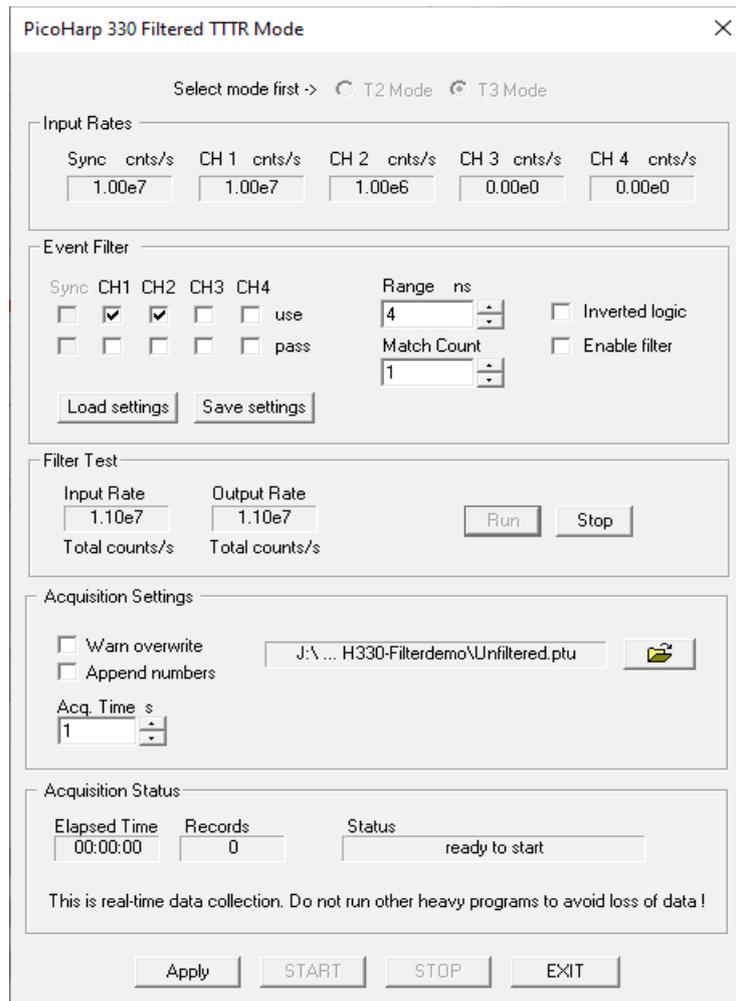
As expected with the filter still being disabled, the Output Rate matches the Input Rate.

After stopping the filter test we can then proceed to a real measurement. This involves the dialog elements for 'Acquisition Settings' and Status as well as the buttons START and STOP. The procedure is the same as in plain TTTR mode described in section 5.3.4. We use an acquisition time of 1 s and the filename 'Unfiltered.ptu'. After completion of the measurement we leave the filtered TTTR mode dialog by clicking EXIT. When the software is back in histogramming mode we load the file we have just collected into the regular histogramming mode display. Note that this requires selecting the file type *.ptu in the file opening dialog. As the collected data is from T3 mode, the software will prompt for which range of the TTTR records it should perform the histogramming. We accept the default setting of using the entire range. The result will be a set of histograms as shown in the next figure.
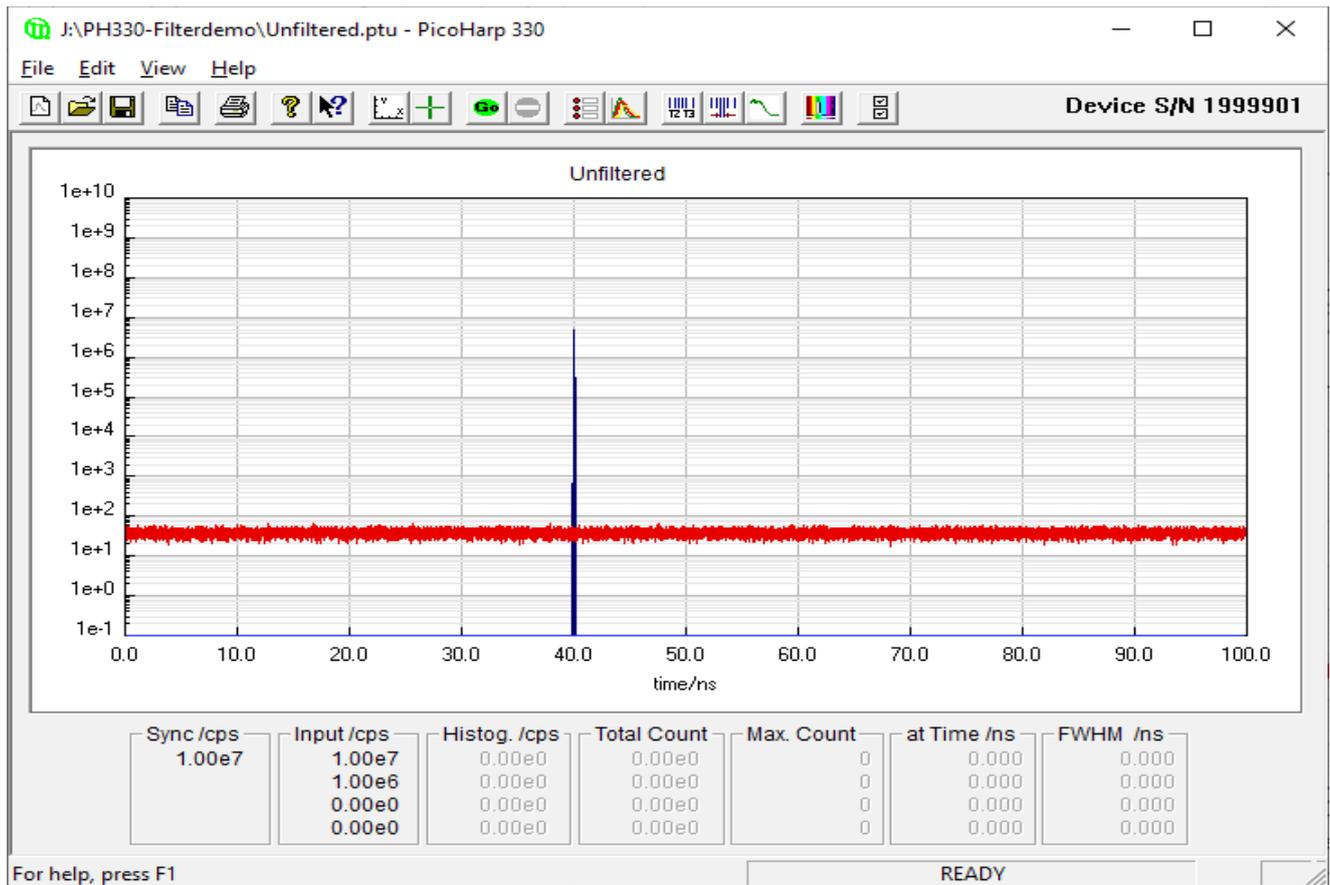
As expected, because the filter was disabled, we obtain histograms exactly as in our initial test in histogramming mode.

Having completed this basic test we can now return to the filtered TTTR mode dialog and proceed to a filtered measurement. The filter settings are the same as previously, except that the filter is now enabled.

Like before, we perform a quick filter test. As we see in the next figure, the output rate has now dropped significantly.
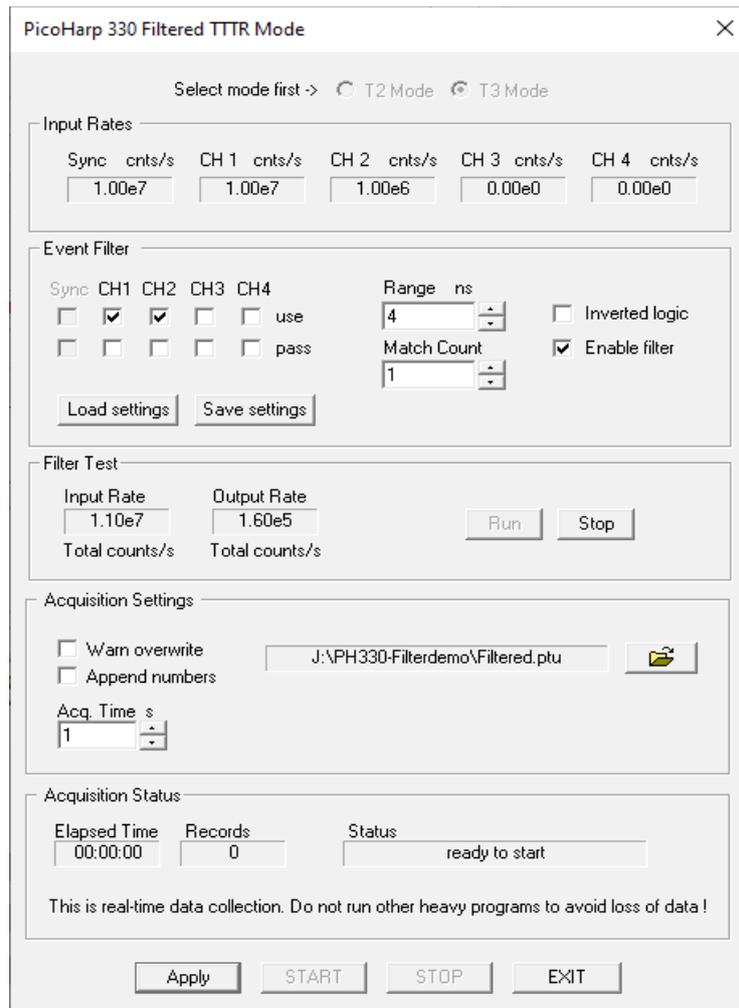
After stopping the filter test we can again proceed and perform a real measurement. This time we name the file 'Filtered.ptu'. After completion of the measurement we can again leave the filtered TTTR mode dialog and then load the new file into the regular histogramming mode display. As this switching between TTTR and histogramming mode always takes time for re-configuration you may want to use a little trick: If you open a second instance of the PicoHarp 330 software you can leave the first instance in TTTR mode and use the second instance as a file viewer. Whichever way you load the file, the result will be a set of histograms as shown in the next figure.
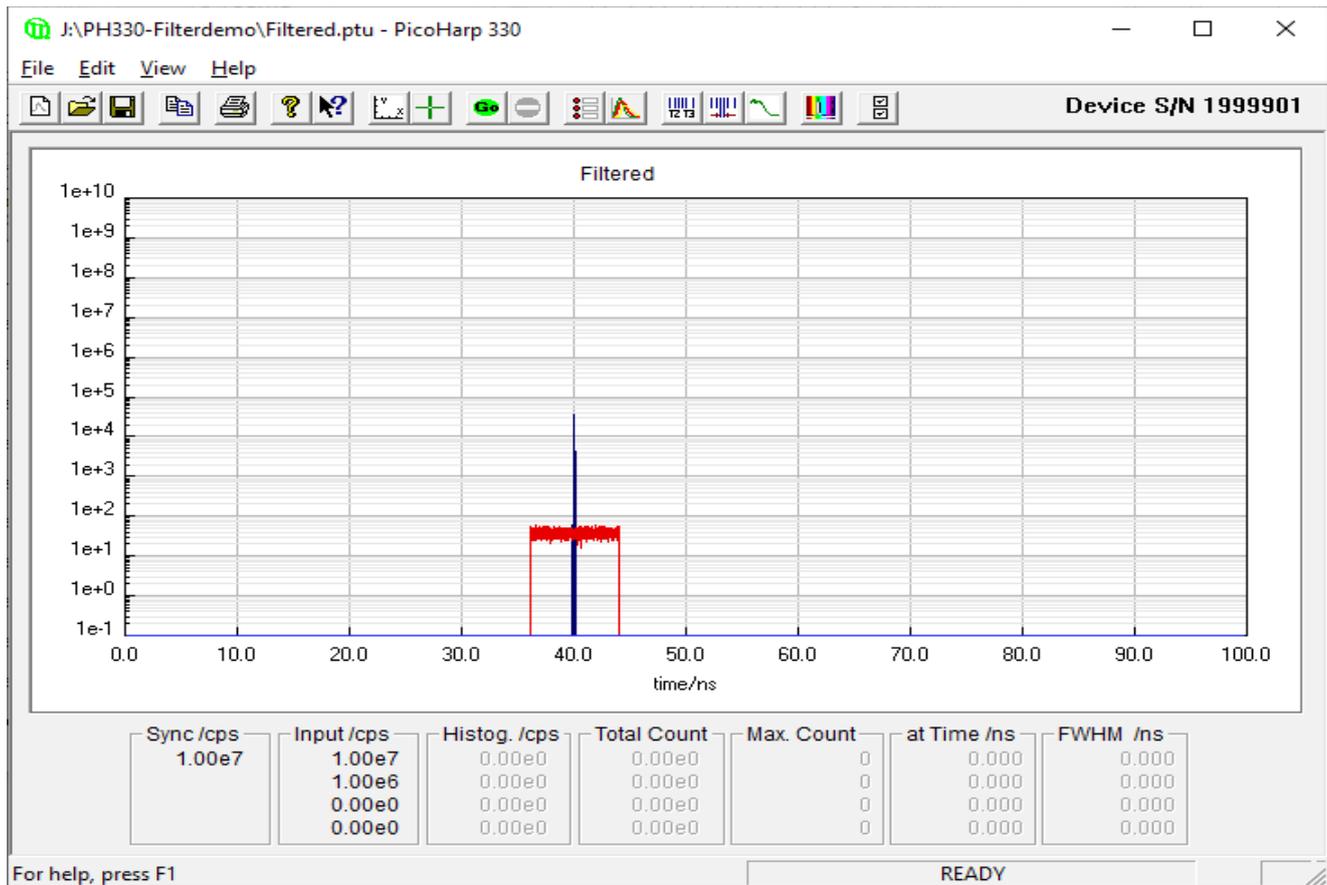
The effect of the filter can now be seen very nicely. First observe the red trace (Generator G2 on Channel 2). The histogram is now empty everywhere, except in the vicinity of the blue peak. The remaining slice of the red trace is exactly the chosen filter range of 4 ns on either side of the blue peak from G1. Next, observe the height blue peak. Compared to the unfiltered data collection it is now substantially smaller. This also is the effect of the filter, as all blue events that had no red companion in the 4 ns range were suppressed. This now gives a nice visual explanation for the significantly lower output rate of the filter we observed before.

Finally, we can perform the same experiment with the filter switched to 'Inverted logic', leaving all other settings as before. This means that the filter will now suppress events that previously would not have been suppressed and vice versa. This inversion may have rather few practical use cases but as you will see shortly, it makes for a nice way of filter verification and demonstration.

The following figure shows the corresponding state of the filter dialog with the filter test running. The output rate is now significantly closer to the input rate but from the fact that it is lower than the input rate we see that some events are still being filtered out. Indeed, you may check that the filter output rates of inverted versus uninverted logic add up to the input rate.
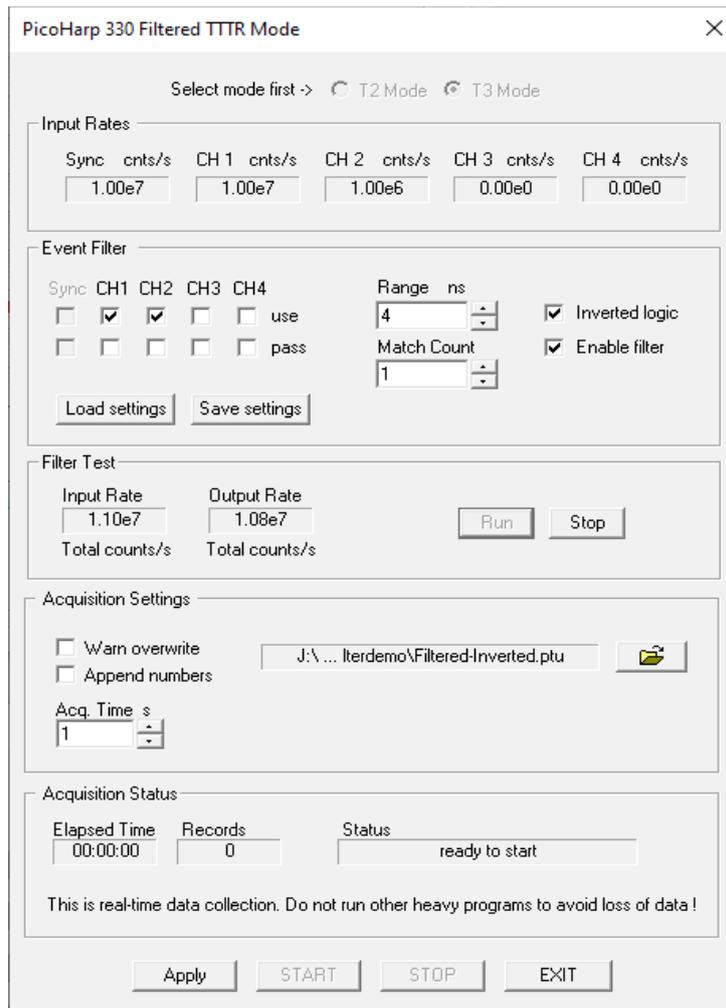
Performing a real measurement into the file 'Filtered-Inverted.ptu' and loading the file back into the histogram display we can nicely visualize the inverse filter operation.
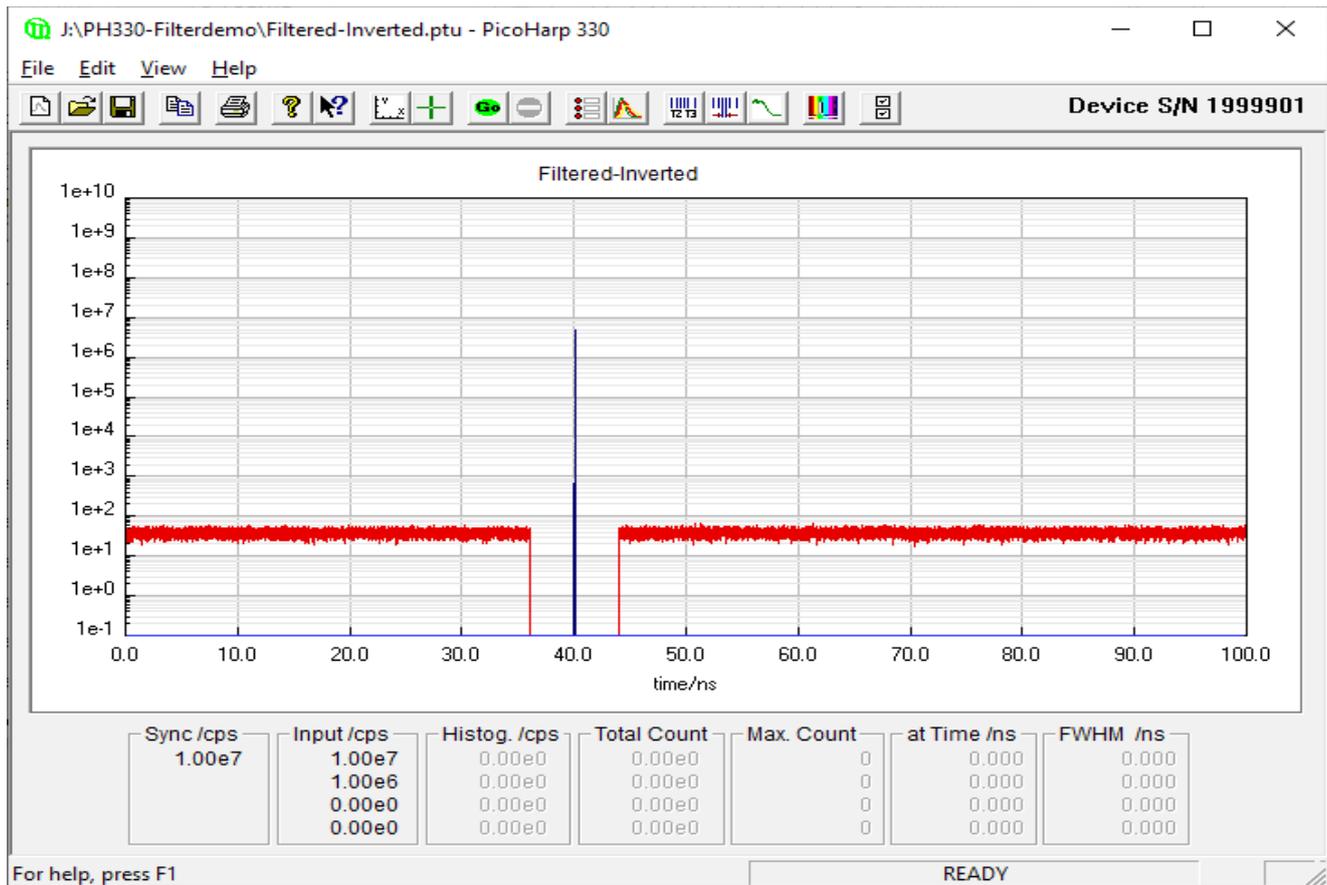
The part of the red histogram that was previously kept in is now punched out and the blue peak is diminished much less, according to the ratio of removed versus unremoved events in the red histogram.

A practical hint: It is possible and quite useful to keep the filter test running while playing with the individual filter settings and observing the output rate on the fly. However, any change of filter settings must always be confirmed with Apply in order to take effect.

All other aspects of TTTR data collection with event filtering are the same as in plain TTTR mode as described in sections 5.3.1 through 5.3.6. Topics such as using external markers or how to use the data files should be looked up there.

5.4. Sequence Mode

In time-resolved fluorescence research it is often of interest to observe time-dependent phenomena. In the simplest case one would then collect a sequence of decay histograms. A similar requirement arises for recording Time-Resolved Excitation/Emission Spectra (TRES), ideally with automated wavelength scan and TCSPC data collection under full software control.

In order to facilitate a generic range of such applications with great flexibilty, the PicoHarp 330 software provides an automated sequence (SEQ) measurement mode. This mode first of all allows software controlled repetition of a histogram mode measurement. For more complex automated experiments it also allows calling a custom piece of software before each measurement. This can be a Windows executable, a batch file or even a Python script. By way of this custom code the users can control their own external hardware, for instance a monochromator. This enables automated collection of spectrally resolved lifetime histograms. Similarly some other parameter such as temperature could be stepped through. SEQ mode data is collected as in standard "Integration Mode" and saved in different blocks of memory for each cycle. Note, however, that SEQ mode data is always collected through input channel 1 only.

SEQ mode can be configured by clicking the corresponding button on the toolbar: ▓
This will launch the dialog for SEQ mode setup.

There you can first of all set up the length of the sequence (the number of measurements) and optionally also the duration of a pause between the measurements.
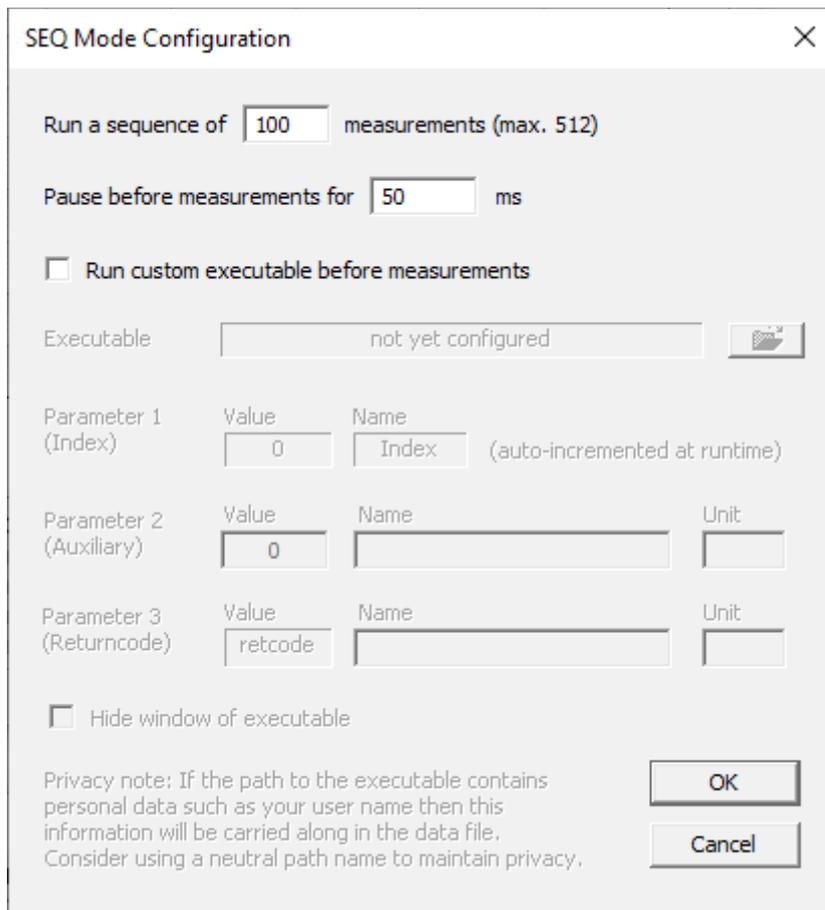
If then below you select (tick) the tickbox "Run custom executable before measurements" you will see further dialog elements getting enabled. The topmost row in this category holds the path name of the executable. It will appear in red if it is not yet configured or invalid. In order to select an executable just click the "file open" button on the right and browse to a suitable file.

The executable, batch file or Python script must be written for the specific purpose and it must follow some rules. The fundamental idea can be illustrated looking at the next three rows of configuration elements called Parameter 1 … Parameter 3, some of which can be edited. Parameters 1 and 2 are going to be passed to the custom executable. Parameter 1 is simply the zero based index of the measurement. It will automatically increment and cannot be edited. Parameter 2 is an auxilliary value that can be edited and will also be passed to the executable. Its value will remain fixed throughout the sequence. This can be used to parametrize the behavior of the executable without having to re-compile it or editing the script. Parameter 3 is the return code from the executable. All three parameters will be written to the header of the corresponding data curve in the SEQ mode file. If, for instance, the executable was built to control a monochromator, it could be designed to return the wavelength of this measurement. By definition a negative return code will indicate an error and the run of the sequence will be aborted. Because of constraints in Windows the return code can only be a 32-bit integer. Demo code fo an executable in C, a batch file and a Python script can be found in the subfolder SEQdemo under the path where you installed the PicoHarp 330 software.

The fields *Name* and *Unit* can be populated with custom strings in order to embed some documentation of the measurements in the file. For similar reasons the path of the executable will be embedded in the file.

Further below in the SEQ configuration dialog you will find another tick box "Hide window of executable". When it is not ticked, the executable's window will pop up each time it is called, i.e. for each step in the sequence. This can be useful for debug purposes but it may also become annyoing. In that case the box for hiding it should be ticked. The executable's window will then not pop up.

Running SEQ Measurements

Note that starting a SEQ measurement will clear and subsequently overwrite all curves in memory that may have been recorded previously. Note also that a SEQ file will remain a SEQ file for its lifetime. This means that after creating or re-loading such a file you can do only two things: perform a new SEQ measurement (effectively overwriting the old file contents) or measure an IRF into curve 0 (overwriting only curve 0). If you want to perform any other measurement you will need to create a new file or load a non-SEQ file.

The SEQ measurement is selected in the PicoHarp 330 control panel. You need to use the selection box for the measurement mode to select SEQ mode (instead of OSC or INT).
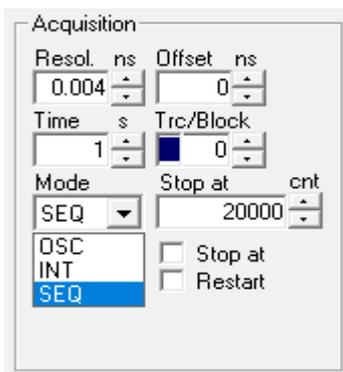
A SEQ measurement can then be started or canceled as usual via the toolbar buttons for start and stop. The data acquisition will be performed like a standard "Integration Mode" measurement, while the block number is incremented for each new wavelength.

At the beginning of each SEQ run, the active curve (block) will be set to 0. All curves that were previously collected and not saved to a file will be overwritten with new data. During a SEQ run, the currently collected data will always be shown as Trace 0 (dark blue), even though the block (Trace) number is actually incremented at each step. This is to overcome the limitation of only 16 display curves being available. You can check the block number in the control panel to see the curve currently being collected. Manual entry will be disabled during the run. Also, you can watch the status bar to see the current activity. Note that for a SEQ measurement the control panel setting 'Stop at' applies as in Integration mode, while 'Restart' is meaningless in SEQ mode.

SEQ Data Analysis and Visualization

Having collected a complete SEQ data set you can save it to a regular `*.phu` file. You can also inspect individual curves via the Trace Mapping dialog. You can furthermore use the software tool FluoPlot from PicoQuant to visualize and analyze the data in various 2D and 3D representations with a multitude of options for coloring, scaling, and changing view aspects in 3D. The figures below show FluoPlot visualizations from a time resolved emission spectra measurement of mixed oxazine dyes in ethanol.
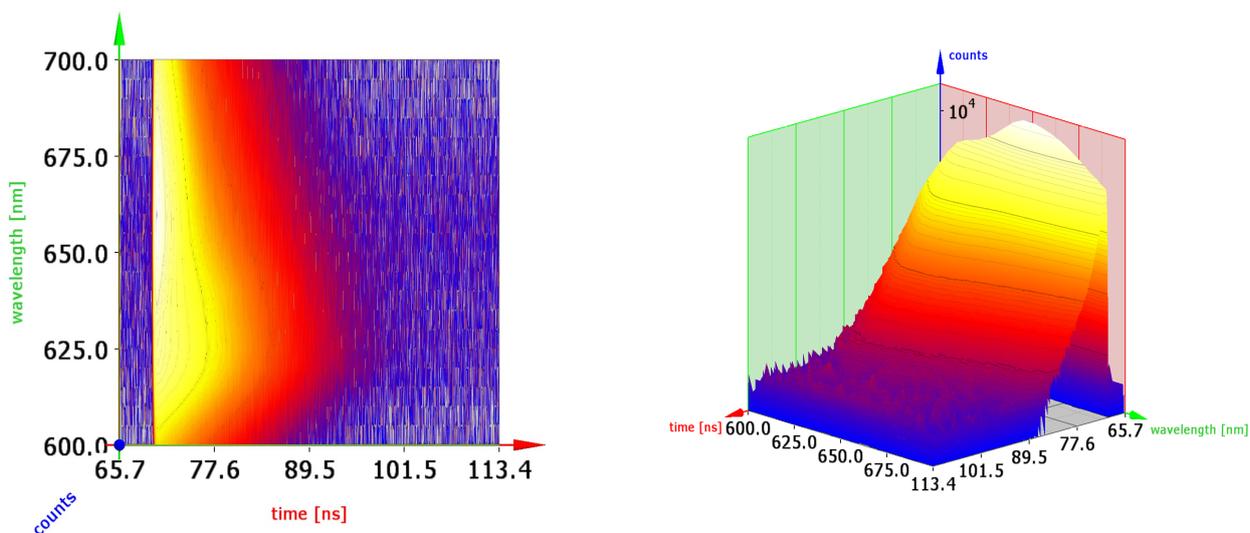
FluoPlot is provided on your software installation media. Just run *FluoPlotSetup.exe* from there. If you received your PicoHarp 330 software by download or email you may need to request FluoPlot separately. Note that the program requires a graphics processing unit with support for OpenGL version 1.5 or higher. Speedy handling of large data files requires sufficient system memory. Further data analysis may need to be performed by dedicated software, either custom programs or specialized solutions available from PicoQuant.

5.5. Multi-Channel Scaling

In contrast to classic TCSPC systems (based on TACs), the sync and the signal channels of the PicoHarp 330 are completely independent. This makes it possible to allow multi-stop measurements, i.e. the detection of multiple photons between two subsequent START signals. The shorter the dead times of the detectors and the timing electronics are, the less these multi-stop measurements are prone to dead-time-induced photon losses. If the histogram bin width is chosen larger than the dead time, the pile-up artifacts due to dead time are entirely eliminated and it is possible to measure at photon rates much higher than the classic pile-up limit of TCSPC. The PicoHarp 330 can then be operated like a Multi-Channel Scaler.

The easiest way to perform multi-channel scaling with the PicoHarp 330 is via histogramming mode. The detector is connected to one of the input channels. In addition, a reference signal at the sync input marking the start of the MCS measurement is needed (see chapter 8.3.1 for the signal specifications).

The integration time should then be set to a value significantly larger than the time bin width multiplied by the number of time bins (65,536). Then start a measurement in in integration (INT) mode. The resulting histogram will display the MCS curve. Please note that the time range that can be obtained with this method is limited to the time bin width multiplied by 65,536. In order to allow large time spans the the time bin width (resolution) must be adjusted as necessary. MCS measurements on longer time ranges are possible via T2 mode. However, this will require additional software for data analysis (e.g., SymPhoTime 64 od snAPI from PicoQuant). Alternatively, if you want to consider writing your own software, you can use The PicoHarp 330 programming library where histogramming mode is supported with up to 524,288 time bins.

6. Controls and Commands Reference

6.1. Main Window

The Title Bar

The title bar is located along the top of a window. To move the window, drag the title bar. Note: You can also move dialog boxes by dragging their title bars. The title bar may contain the following elements:

System Menu	
PicoHarp 330 – [Name of the file or dialog]	
Minimize Button	▬
Maximize or *Restore* Button, resp.	▢ / ▤
Close Button	✕

Scroll bars

Displayed at the right and bottom edges of the PicoHarp 330 window if a certain minimum window size is reached. The scroll boxes inside the scroll bars indicate your vertical and horizontal location in the display area. You can use the mouse to scroll to other parts of the window.

Size command

Use this command to resize the active window.

Note: The command is unavailable for already maximized or minimized windows.

> **Shortcut**
>
> Mouse: Drag the corners or edges of the window.
> Keys: `<Alt>+<Space> S`

Minimize command

Use this command to reduce the PicoHarp 330 window to an icon. Running measurements will continue.

Note: The command is unavailable for already minimized windows.

> **Shortcut**
>
> Mouse: Click the minimize button ▬ on the title bar.
> Keys: `<Alt>+<Space> N`

Maximize command

Use this command to enlarge the active window to fill the available space.

Note: The command is unavailable for already maximized windows.

> **Shortcut**
>
> Mouse: Click the maximize button ▢ on the title bar;
>   or double–click the title bar of a non-maximized window.
> Keys: `<Alt>+<Space> X`

Restore command

Use this command to return from a maximized or minimized window to the previous size.

Note: The command is only available for already maximized or minimized windows.

   Shortcut

   Mouse: Click the maximize button ⬒ on the title bar;
                       or double–click the title bar of the enlarged window.
   Keys: `<Alt>+<Space> R`

Close command

Use this command to close the active window or dialog box.

Double–clicking a system menu box is the same as choosing the Close command.

   Shortcuts

   Mouse: Click the close button ⨯ on the title bar.

   Keys: `<Alt>+<F4>`
                       `<Alt>+<Space> C`

Panel Meters

At the bottom of the main window there is a set of panel meters. The meters showing units of cps (counts per second) in their title are rate meters. The leftmost rate meter shows the sync input rate. The meter next to it shows the channel input rates. The other meters show the histogramming rate, the total count in the histogram, the maximum (peak) count, the position of the maximum and the full width at half maximum (FWHM) of the histogram peak, respectively.

Sync /cps	Input /cps	Histog. /cps	Total Count	Max. Count	at Time /ns	FWHM /ns
1.01e6	1.01e6	1.01e6	1.01e6	70289	5.871	0.005
	1.01e6	1.01e6	1.01e6	68260	5.989	0.006
	1.01e6	1.01e6	1.01e6	68050	5.980	0.006
	1.01e6	1.01e6	1.01e6	69169	5.901	0.006

Note that the rate meters have a fixed gate time of 100 ms and limited accuracy, notably at low rates. They are only intended as means of quick diagnostics and should not be taken for actual measurements. The panel meters can be enlarged by double-clicking them, which is useful when performing optical alignment or similar tasks when the PC monitor is some distance away.

6.2.   Menus

6.2.1.   File Menu

The *File* menu offers the following commands:

New	Clears all histogram data and restores default settings.
Open	Opens an existing histogram file.
Save	Saves a histogram file.
Save As…	Saves an opened histogram file to a specified file name.
Print	Prints the currently displayed histogram.
Page Setup	Allows modifying the page layout for printing.
Print Preview	Displays the layout as it would appear printed.
Print Setup	Selects a printer and printer connection.
1…4 <Recent Filename>	Opens one of the four last recently opened files
Exit	Exits the PicoHarp 330 software.

New command

Use this command to create a blank histogram with the last default settings. You can open an existing histogram file with the *Open* command.

> **Shortcuts**
>
> Toolbar:
>
> Keys: `<Ctrl>+N`
>   `<Alt>+F N`

Open command

Use this command to open an existing histogram file. All control panel settings will also be restored. You can also open PicoHarp 330 histogram files (`*.phu`) by double clicking them or dragging them onto the PicoHarp 330 icon. You can revert to a blank histogram and default control panel settings with the New command. You can also select `*.ptu` files for loading as histogram data but they must be T3 mode files. Loading such files as histogram data is for diagnostics only and cannot make use of their full information content.

> **Shortcuts**
>
> Toolbar:
>
> Keys: `<Ctrl>+O`
>   `<Alt>+F O`

File Open dialog box

The following options allow you to specify which file to open:

Look in: Select (i.e. browse into) the directory in which the file that you want to open resides.

Main box   In this box you see the content of the chosen directory, filtered by *Files of type.*

File name: Type or select the filename you want to open. This box lists files with the extension you select in the *Files of type* box.

Files of type   Select the type of file you want to open.

Save command

Use this command to save the current histogram data to a file with current name and directory. When you save a histogram for the first time, PicoHarp 330 displays the *Save As…* dialog box so you can name your file. This command is unavailable if a measurement is running. If you want to change the name and directory of an exist-ing file before you save it, choose the *Save As* command.

> **Shortcuts**
>
> Toolbar: 💾
>
> Keys: `<Ctrl>+S`
>   `<Alt>+F S`

Save As… command

Use this command to (re–)name the current histogram data and save. The software displays the *Save As…* dia-log box so you can name your file. To save a file with its existing name and directory, use the Save command.

> **File Save As… dialog box**
>
> The following fields allow you to specify the name and location of the file you are about to save:
>
> *Save in:*   Select (i.e. browse into) the directory where you want to store the file.
>
> *Main box*   In this box you see the content of the chosen directory, filtered by *Save as type.*
>
> *File name:*   Type a new filename to save a histogram with a different name. The software automati-cally adds the extension you specify in the *Save as type:* box.
>
> *Save as type:* Selects a filter on the directory chosen in the *Save in:* box. Only files that pass the filter are shown in the dialogs main box. Additionally the software adds the extension associ-ated with the chosen filter to a given file name if it was entered without extension.

Print command

Use this command to print the currently displayed histogram curves. This command presents a Print dialog box, where you may specify the number of copies, the destination printer, the paper orientation and other printer setup options.

> **Shortcuts**
>
> Toolbar: 🖨
>
> Keys: `<Ctrl>+P`
>   `<Alt>+F P`

Page Setup command

Use this command to change the print layout. This command presents the page setup dialog box, where you may select or deselect various items to appear in the print. You can use print preview to check the resulting lay-out.

> **Shortcut**
>
> Keys: `<Alt>+F U`

Print Preview command

Use this command to display the layout as it would appear when printed. The main window will be replaced with a print preview window in which one or two pages will be displayed in their printed format. The print preview toolbar offers you options to view either one or two pages at a time; move back and forth through the document; zoom in and out of pages; and initiate a print job.

> **Shortcut**
>
> Keys: `<Alt>+F V`

Print Setup command

Use this command to select a printer and a printer connection. This command presents a Print Setup dialog box, where you specify the printer and its connection.

> **Shortcut**
>
> Keys: `<Alt>+F R`

1, 2, 3, 4 command (most recently used files)

Use the numbers and filenames listed at the bottom of the *File* menu to open the last four files you closed. Choose the number that corresponds with the file you want to open.

> **Shortcuts**
>
> Keys: `<Alt>+F 1`
>   `<Alt>+F 2`
>   `<Alt>+F 3`
>   `<Alt>+F 4`

Exit command

Use this command to end your PicoHarp 330 session. Save your data before exiting.

> **Shortcuts**
>
> Mouse: Click the close button ☒ on the title bar.
> Keys: `<Alt>+<F4>`
>   `<Alt>+F X`

6.2.2.   Edit Menu

The *Edit* menu offers only one command:

Copy	Copies the displayed curves.

Copy command

Copies the currently displayed curves in ASCII format to the clipboard. This can be used to export histogram data to spreadsheet or data analysis software, e.g., the "EasyTau 2" Software. Note that only the currently selected curves within the current display limits are copied. If the curves in the display are of different resolution then all time bins will be copied. Copying data to the clipboard replaces the contents previously stored there.

The exported data is plain `<Tab>` separated ASCII and can be read by most commonly used spreadsheet programs or data analysis software.

This command is unavailable while a measurement is running.

> **Shortcuts**
>
> Keys: `<Ctrl>+C`
>   `<Alt>+E C`

6.2.3. View Menu

The View menu offers the following commands:

Toolbar	Shows or hides the toolbar.
Status Bar	Shows or hides the status bar.
Axis Panel	Shows the axis settings panel.
Control Panel	Shows the control panel.
Trace Mapping	Shows the trace mapping dialog.

Toolbar command

Use this command to display and hide the Toolbar, which includes buttons for the most common commands, such as *File Open*. A check mark appears next to the menu item when the Toolbar is displayed.

> **Shortcut**

> Keys: `<Alt>+V T`

Status Bar command

Use this command to display or hide the status bar at the bottom of the PicoHarp 330 main window. The left pane of the status bar describes the action to be executed by a menu item or toolbar button as the mouse pointer hovers over it. The right hand side of the status bar shows the current measurement activity and key- board latch state. A check mark appears next to the menu item when the status bar is displayed.

> **Shortcut**

> Keys: `<Alt>+V S`

Axis Panel command

Use this command to display the axis settings panel. The result is the same as the axis panel button in the tool- bar or double clicking on the histogram axes. The axis settings panel can be closed by clicking its close button.

> **Shortcut**

> Keys: `<Alt>+V A`

Control Panel command

Use this command to display the control panel. This command has the same effect as the control panel button in the toolbar. The control panel can be closed by clicking its close button.

> **Shortcut**

> Keys: `<Alt>+V C`

Trace Mapping command

Use this command to display the trace mapping dialog. This command has the same effect as the correspond- ing button in the toolbar. The trace mapping dialog can be closed by clicking its close button.

> **Shortcut**

> Keys: `<Alt>+V M`

6.2.4. Help Menu

The *Help* menu offers the following commands, which provide you assistance with this application:

Help Topics	Offers you the contents list of topics on which you can get help.
Help Index	Offers you an index to topics on which you can get help.
Help Search	Offers you a means of full text search on all help topics.
Activate Context Help	Switches the cursor mode to "point to item" for context help.
Check for Updates	Checks for the availability of a newer software version.
Visit Website	Opens www.picoquant.com in your browser.
Request Support	Provides support details and access to the support website.
About PicoHarp 330 330…	Displays version information of the PicoHarp 330 software.

Note: Online help (context help) on most functions, dialogs, control items etc. is available via the F1 key.

Help Topics command

This command opens a browser offering a tree view like list of contents, and a help page. Browse through the contents with your mouse or the cursor keys. `<Enter>` opens / closes chapters, `<Cur Up>`/`<Cur Down>` turns the pages. The help page to the right of the browser will always be held up–to–date.

> **Shortcuts**
>
> Keys: `<F1>`   (if online help was last recently opened in topics mode)
>   `<Alt>+H T`

Help Index command

This command opens a browser offering an index list of keywords and a help page. Browse through the contents with your mouse or the cursor keys. `<Enter>` opens the current item, `<Cur Up>`/`<Cur Down>` navigates through the index.

> **Shortcuts**
>
> Keys: `<F1>`   (if online help was last recently opened in index mode)
>   `<Alt>+H I`

Help Search command

This command opens a browser offering full text search on words or word groups over the whole online help volume. You can enter more specific search pattern, using the operators "*AND*", "*OR*", "*NEAR*" and "*NOT*".

> **Shortcuts**
>
> Keys: `<F1>`   (if online help was last recently opened in search mode)
>   `<Alt>+H S`

Activate Context Help

Changes the mouse cursor mode to context help mode. You can obtain context sensitive help for a visual element in the GUI by simply clicking on it.

> **Shortcuts**
>
> Keys: `<Shift>+<F1>`
>   `<Alt>+H C`

Check for Updates command

Use this command to check if a newer software version is available. This requires internet access. If a newer version is available it can immediately be downloaded. Note that this does not perform any installation. You still need to unzip and install the downloaded software. Do this only after reading the release notes.

> **Shortcut**
>
> Keys: `<Alt>+H U`

Visit Website command

Use this command to open www.picoquant.com in your browser. Of course this requires internet access.

> **Shortcut**
>
> Keys: `<Alt>+H W`

Request Support command

Use this command to open a form with important support details of your system and direct access to the support page on the web. The latter requires internet access. If you cannot access the web directly, please copy the support details and send them later by email to support@picoquant.com together with a precise description of the problem. If your setup is not working at all, provide at least **the serial number of your PicoHarp 330** and a precise description of the installation environment and the observed issues, including the exact wording of any occurring error messages.

> **Shortcut**
>
> Keys: `<Alt>+H S`

About PicoHarp 330 ... command

Use this command to display the copyright notice and version number of your PicoHarp 330 software and hardware, if installed. It also provides access to the PicoQuant Web site and software updates.

> **Shortcut**
>
> Keys: `<Alt>+H A`

6.3. Toolbar

Toolbar buttons of special interest are explained in some more detail here:

Context Help

When you choose the *Context Help* button, the mouse pointer will change to an arrow with question mark. If you then click somewhere in the PicoHarp 330 window, such as on another toolbar button, the help topic associated with it will be shown.

Axis Panel

Clicking this button opens the *Axis Panel*. This panel provides controls for the axis limits and allows to switch between lin / log scaling. Double-clicking the axes opens the same dialog. The dialog is non-modal, so it can remain permanently open. The panel will remember its last position when closed. When it is opened the next time it will be at that screen position again.

Data Cursor

Opens the data cursor dialog box. You can use this dialog to mark and retrieve the data values of individual histogram bins. A pair of crosshairs will be provided for marking data points with the mouse or keyboard. When the data cursor dialog is activated, the crosshairs will appear in the histogram display area. By clicking on or near a data point, the current crosshair (black) will snap to that point. At the same time a second crosshair (grey) will jump to the previously marked point. The data cursor dialog then shows the current count and time values for these points as well as the corresponding differences (deltas). The data cursor always applies to one "active" curve only. You can select this active curve at the top of the dialog where it is shown with its corresponding curve color. While the grey crosshair normally jumps to the previous data point, you can modify this behavior by holding down the SHIFT key while clicking on curve points. The grey crosshair will then remain at its previous position. This allows finding a particular point while keeping the other one fixed.

Another advanced mode of operation of the data cursor is possible via the LEFT and RIGHT arrows of the keyboard. Pressing these keys will direct the black cursor to the next left or right point in the curve. Again, the behavior of the marker for the previous point (grey cross) can be controlled via the SHIFT key.

Start Button

Starts a histogram measurement based on the current PicoHarp 330 control panel.

Stop Button

Stops a histogram measurement interactively. Note that histogramming is performed in software from a TTTR mode data stream. If the count rates are very high the device FIFO may run full in this process. The data remaining in the FIFO still gets processed after the measurement is stopped. It may therefore happen that there is some lag time in the response to clicking this button.

Control Panel

Launches the PicoHarp 330 control panel. If the control panel is already open, subsequent clicking of this button will just make the control panel the active window. The dialog is non-modal, so it can remain permanently open. The panel will remember its last position when closed. When it is opened the next time it will be at that screen position again, even in the next PicoHarp 330 session.

Trace Map

Launches the *Trace Mapping* dialog. The PicoHarp 330 software can measure histograms in up to 512 memory blocks. Out of these, up to 16 curves can be displayed. Displaying more traces at a time would result in too much clutter on the screen. The trace map dialog is used to select the curves to display. Tick the individual "S*how*" boxes to display a curve. Select the number of the memory block you wish to map the individual display curve to. Choose the active memory block you wish to use for the next measurement in the PicoHarp 330 control panel.

TTTR Mode

Opens the TTTR mode dialog box. Use this dialog to enter the acquisition time and the destination file for a TTTR mode run. Make sure all other measurement parameters have been set and tested in interactive mode before entering TTTR mode.

TTTR Real–Time Correlator

Opens the TTTR mode real-time correlator dialog box. Use this dialog for FCS preview during a TTTR mode run. Make sure all other measurement parameters have been set and tested in interactive mode before entering TTTR correlator mode. Changing to this mode takes a few seconds for reconfiguration of the hardware. See also section 5.3.7.

SEQ Mode Configuration

In order to measure sequences of histograms the PicoHarp 330 software provides a SEQ measurement mode, that allows to call custom code to control custom hardware. This could be a monochromator controlled via stepper motors for automated collection of spectrally resolved lifetime histograms. Clicking this button will launch the dialog for SEQ mode setup. There you can set parameters such as the sequence length, a pause time and the custom code to be called. Demo code fo an executable in C, a batch file and a Python script can be found in the subfolder *SEQdemo* under the same path where the PicoHarp 330 software was installed.

General Settings

Opens the software settings dialog box. Use this dialog to change standard settings of the PicoHarp 330 software. Notably these are: *Display rate* (0.1 to 1s), *Draw mode* (lines, stairs), *Grid* check box, *Prompt overwrite* (warning before overwriting existing data), selective disabling of *Warnings* and *TTTR Marker Settings.* The control connector of the PicoHarp 330 provides TTL inputs for synchronization signals. The markers can be enabled or disabled as well as recorded at either the rising or falling edge of the corresponding TTL signal. The active edges can be chosen here. There is also a programmable hold-off time to suppress glitches on the marker signals. The dialog also allows selective disabling of warnings that you do not wish to receive. All settings will be kept in the Windows registry and will be retrieved at the next program start. They are stored individually for each Windows user profile.

6.4.   Control Panel

The control panel consists of several sections containing edit boxes and other controls for related parameters. These are described in the following subsections. Note that the settings of the control panel as well as the position of the control panel on the screen will be stored in the registry and retrieved at the next program start. When you load a PicoHarp 330 data file the settings of the control panel will change according to the settings in that file. This can be used as a means of reverting to standard settings for new experiments.

6.4.1.   Trigger Out

The settings in this group configure the behavior of the trigger output. This is essentially a programmable signal generator delivering negative pulses through a SMA connector at the back of the PicoHarp 330. It can be used to trigger a light source that has no clock of its own.

Trigger Output Period   edit box and spin control

Here the period of the trigger output can be set. The unit is microseconds. The allowed range is 0.1 μs to 1.6 s.

Trigger Output Force On   check box

Here the trigger output can be switched on regardless of whether a measurement is running.

Trigger Output Auto On   check box

Here the trigger output can be set to turn on only when a measurement is running. This can reduce bleaching of light sensitive samples. Note, however, that some light sources may change their pulse shape due to warming-up after a longer off-period.

6.4.2.   Sync–Input

The settings in this group configure the behavior of the Sync input. See section 5.1 "Setting Up the Input Channels" to get started.

EdgeTr. / CFD   radio buttons

Here the sync trigger input circuit can be configured for Edge Trigger versus Constant Fraction Discriminator (CFD). The latter is useful for detectors with fluctuating pulse height but allows only negative going pulses.

when above EdgeTr. is selected:

Edge   edit box and spin control

Here the trigger edge (rising/falling) for the Sync input can be set. The value 0 stands for falling, 1 stands for rising edge. Type the value as an integer in the edit box and press `<Enter>` or click on *Apply*. Alternatively, use the spin control (next to it) to increment / decrement the current value. In this case changes take effect immediately without pressing `<Enter>`.

Level   edit box and spin control

Here the trigger level for the Sync input can be set. Units are millivolts (mV), the permitted range is -1500 to 1500. Type the value as an integer in the edit box and press `<Enter>` or click on *Apply*. Alternatively, use the spin control (next to it) to increment / decrement the current value. In this case changes take effect immediately without pressing `<Enter>`.

when above CFD is selected:

ZeroCr.   edit box and spin control

Here the zero cross for the Sync CFD can be set. Units are millivolts (mV), the permitted range is -100 to 0. Type the value as an integer in the edit box and press `<Enter>` or click on *Apply*. Alternatively, use the spin control (next to it) to increment / decrement the current value. In this case changes take effect immediately without pressing `<Enter>`.

Level   edit box and spin control

Here the discriminator level for the Sync input can be set. Units are millivolts (mV), the permitted range is -1500 to 0. Type the value as an integer in the edit box and press `<Enter>` or click on *Apply*. Alternatively, use the spin control (next to it) to increment / decrement the current value. In this case changes take effect immediately without pressing `<Enter>`.

Offset   edit box and spin control

Shifts the relative timing of Sync and photon events and is designed to compensate optical and electrical delays (due to differences in cable lengths and optical paths), e.g., in order to shift your fluorescence decay within the sync frame so that it is not truncated. This feature completely eliminates the need for adjustable delay boxes.

Units are picoseconds (ps), the permitted range is from $-99,999$ to $+99,999$ ps. Type the value as an integer in the edit box and press `<Enter>` or click on *Apply*. The offset value can be entered in steps of picoseconds. Alternatively, use the spin control next to the edit box to increment/decrement the current value. In this case changes take effect immediately without pressing `<Enter>`.

Tdead   edit box and spin control

Tdead is a programmable dead time for suppression of afterpulsing artefacts of some detectors. On the sync input it is rarely needed. Leave it at 0 if you work with a sync signal from a laser.

Units are nanoseconds (ns), the permitted range is from 0 to 160 ns. Note that Tdead = 0 does not really mean zero deadtime, instead, the input will then operate at its shortest (native) dead time of about 650 ps. When a dead time >0 is set then the timing hardware will suppress events falling into the dead time. This is only a suppression of further processing and does not prevent the TDC to go into another (native) dead time.

Type the value as an integer in the edit box and press `<Enter>` or click on *Apply*. The value can be entered in steps of nanoseconds. Alternatively, use the spin control next to the edit box to increment/decrement the current value. In this case changes take effect immediately without pressing `<Enter>`.

Divider   edit box and spin control

Here the programmable divider of the Sync input can be set. This allows to reduce the sync input rate so that the front end FIFOs can handle it. This is mandatory for fast sync sources >81 MHz. At very high overall rates on all channels it may be necessary to set a divider >1 even if the sync rate is not >81 MHz. This is because the front end data processing may then reach its limits and sync events may get lost. This manifests itself in photon events appearing outside the expected sync period. Otherwise set the divider only as large as necessary. In any case internal logic determines the sync period and re-calculates the sync events that were divided out. It should be noted that this only works with stable sync sources that provide a constant pulse-to-pulse period. All fast laser sources known today meet this requirement within an error of a few picoseconds. With random and low rate signals (< 1 MHz), the divider must be set to 'None'.

6.4.3.   Inp.Chan 1 .. N

The settings in these dialog tabs configure the behavior of the input channels. See section 5.1 "Setting Up the Input Channels" to get started. The PicoHarp 330 can have up to four usable input channels, dependent on how many were purchased. Upgrades can be purchased later. Dependent on how many usable channels the device has, some control elements for the channels may remain disabled. The specific controls for each channel are described below.

EdgeTr. / CFD   radio buttons

Here the trigger input circuit of the channel can be configured for Edge Trigger versus Constant Fraction Discriminator (CFD). The latter is useful for detectors with fluctuating pulse height but allows only negative going pulses.

when above EdgeTr. is selected:

> **Edge**   edit box and spin control
>
> Here the trigger edge for the input channel can be set. The value 0 stands for falling, 1 stands for rising edge. Type the value as an integer in the edit box and press <Enter> or click on *Apply*. Alternatively, use the spin control (next to it) to increment / decrement the current value. In this case changes take effect immediately without pressing <Enter>.
>
> **Level**   edit box and spin control
>
> Here the trigger level for the respective input channel can be set. Units are millivolts (mV), the permitted range is -1500 to 1500. Type the value as an integer in the edit box and press <Enter> or click on *Apply*. Alternatively, use the spin control (next to it) to increment / decrement the current value. In this case changes take effect immediately without pressing <Enter>.

when above CFD is selected:

> **ZeroCr.**   edit box and spin control
>
> Here the zero cross level for the input channel can be set. Units are millivolts (mV), the permitted range is -100 to 0. Type the value as an integer in the edit box and press <Enter> or click on *Apply*. Alternatively, use the spin control (next to it) to increment / decrement the current value. In this case changes take effect immediately without pressing <Enter>.
>
> **Level**   edit box and spin control
>
> Here the discriminator level for the respective input channel can be set. Units are millivolts (mV), the permitted range is -1500 to 0. Type the value as an integer in the edit box and press <Enter> or click on *Apply*. Alternatively, use the spin control (next to it) to increment / decrement the current value. In this case changes take effect immediately without pressing <Enter>.

Offset   edit box and spin control

Shifts the relative timing of photon events and is designed to compensate optical and electrical delays (due to differences in cable lengths and optical paths), e.g., in order to shift your fluorescence decay within the sync frame so that it is not truncated. This feature completely eliminates the need for adjustable delay boxes.

Units are picoseconds (ps), the permitted range is from −99,999 to +99,999 ps. Type the value as an integer in the edit box and press <Enter> or click on *Apply*. The offset value can be entered in steps of picoseconds. Alternatively, use the spin control next to the edit box to increment ∕ decrement the current value. In this case changes take effect immediately without pressing <Enter>.

Tdead   edit box and spin control

Tdead is a programmable dead time for suppression of afterpulsing artefacts as caused by some detectors (e.g. SPADs). Otherwise it is rarely needed. Leave it at 0 unless you have good reasons to set it higher.

Units are nanoseconds (ns), the permitted range is from 0 to 160 ns. Note that Tdead = 0 does not really mean zero dead time, instead, the input will then operate at its shortest (native) dead time of about 680 ps. When a dead time >0 is set then the timing hardware will suppress events falling into the dead time. This is only a suppression of further processing and does not prevent the TDC to go into another (native) dead time.

Type the value as an integer in the edit box and press `<Enter>` or click on *Apply*. The value can be entered in steps of nanoseconds. Alternatively, use the spin control next to the edit box to increment/decrement the current value. In this case changes take effect immediately without pressing `<Enter>`.

6.4.4. Acquisition

Resolution   edit box and spin control

Use this set of input controls to specify the time resolution (time bin width). Units are picoseconds (ps). Possible choices are the device's base resolution and successive multiples by two. Type the desired resolution value as an integer in the edit box and press `<Enter>` or click on *Apply*. Alternatively, use the spin control (next to it) to increment / decrement the current value. In this case, changes take effect immediately without pressing `<Enter>`. In case of entering values other than valid resolutions, the next suitable resolution step is chosen automatically.

There are always $2^{16} = 65536$ time bins in one histogram. With the chosen resolution, the respective time range covered is `65536 * Resolution`. The choice you make must be a compromise between resolution and time span covered. The smallest range offers the best resolution and the shortest span (vice versa for the largest range). For high sync rates the highest resolution is usually most appropriate, since the smallest range still covers the full sync period. For lower sync rates the histogram range may be too small to cover the full sync period. The decay curve region of interest may therefore lie outside the acquisition window of 65536 time bins. Apart from switching to a lower resolution it is possible to shift the acquisition window relative to the sync frame by means of the *offset*. Note that working with very long time spans (low sync rates) at high count rates also requires long acquisition times to minimize noticeable "steps" in the acquired histograms.

Acquisition Time   edit box and spin control

Set the desired measurement time here. Units are seconds (s). The permitted range is 0.001 to 360,000 s in steps of 0.001 s. Type the desired value in the edit box and press `<Enter>` or click on *Apply*. Alternatively, use the spin control (next to it) to increment / decrement the current value. In this case, changes take effect immediately without pressing `<Enter>`. The default increment / decrement per mouse click is logarithmic.

Offset   edit box and spin control

For high sync rates the highest resolution is usually most appropriate, since the smallest range still covers the full sync period. The offset should in these cases always be set to 0.

For lower sync rates the histogram range may be too small to cover the full sync period. The measurement region of interest may therefore lie outside the acquisition window of 65536 histogram bins. Apart from switching to a lower resolution it is possible to shift the acquisition window relative to the sync frame by means of setting the offset > 0.

Use this set of input controls to specify the desired offset. Units are nanoseconds (ns). Possible offsets are in the range from 0 to 100,000,000 ns. Type the value as an integer in the edit box and press `<Enter>` or click on *Apply*. The offset value can be entered in steps of nanoseconds but will internally be rounded to the nearest multiple of the device's base resolution. Alternatively, use the spin control next to the edit box to increment/decrement the current value. In this case changes take effect immediately without pressing `<Enter>`.

Internally the offset is subtracted from each start–stop measurement before it is used to address the histogram bin to be incremented. Therefore, increasing the offset means shifting the signal on the screen to the left, towards earlier times.

Note that the offset has no effect on the relative timing of laser pulses and photon events. It merely shifts the region of interest where data is to be collected. The relative timing of laser pulses and photon events can only be controlled by means of cable delays and the input offsets described in sections 6.4.2 and 6.4.3. The latter may be necessary in order to shift your fluorescence decay within the sync frame so that it is not truncated.

Trc./Block   trace color indicator, edit box and spin control

The PicoHarp 330 software can measure and store histograms in up to 512 memory blocks. Out of these, up to 16 curves can be displayed and one "active block" can be used for measurements. Choose the active memory block you wish to use for the next measurement here. The trace mapping dialog is used to select the curves for display. Make sure the curve you are using is switched on (*shown*). You can open this dialog directly by clicking

on the trace color indicator. The color indicator shows the trace color the chosen block is currently mapped to. If it is a solid square, the curve is mapped and shown. If it is mapped but not shown, the indicator shows a small striped square. If the curve is not even mapped for display the indicator remains white. Make sure not to over-write existing data in a block that was used before. In order to warn you that a trace is used, the heading of the Trc/Block selector will turn red.

Mode   selection box

Here you can select between three different acquisition modes:

OSC (Oscilloscope Mode)

In oscilloscope mode the acquisition and display, once started, repeats at intervals given by the current ac-quisition time setting. The histogram accumulation always starts from scratch. This is useful for monitoring fast changes during optical alignment etc. Note that in this mode you may see nothing for a long time if you set the acquisition time to a high value. Typically one would not set an acquisition time of more than 1 sec-ond in this mode.

INT (Integration Mode)

As opposed to oscilloscope mode, the histogram acquisition in integration mode is not reset to zero with each display refresh. The histogram continues to grow while the display is updated every 0.1 to 1 seconds. The update rate is determined by the refresh rate value selected in the *General Settings Dialog*. Acquisition can be manually started and stopped, additionally the option "*Stop at*" will cause the acquisition to stop when the chosen stop count is reached in any histogram bin. The option "*Restart*" will cause the acquisition to start again after the acquisition time has elapsed. This is similar to oscilloscope mode but allows watching the histogram grow if the acquisition time is relatively long.

SEQ (Sequence Mode)

Selects the measurement mode SEQ for collection of a whole sequence of histograms with an optional call to custom code for hardware control before each measurement. See section 5.4.

Stop at   edit box and spin control

Here the stop count level for histogramming can be set. The setting will only apply if the corresponding check box below is ticked. Type the value as an integer in the edit box and press `<Enter>` or click on *Apply*. Alterna-tively, use the spin control (next to it) to increment / decrement the current value. In this case changes take ef-fect immediately without pressing `<Enter>`. The minimum is 0. If the entered value exceeds the allowed maxi-mum of 4,294,967,295, an error message appears. When using the spin control, exceeding the maximum is not possible.

Stop at   check box

If this box is ticked (checked) the measurement will stop when any histogram bin has reached the '*Stop at*' count level selected right above.

Restart   check box

If this box is ticked (checked) the measurement will automatically restart after the acquisition time has elapsed. Toggle the current setting with a mouse click. The setting is without effect in SEQ mode.

6.5. Axis Panel

The controls of the *Axis Panel* are used to customize the main window's histogram display. All mouse actions on the spin controls on this dialog result in instant modification of the display. If you choose to alter the values in the edit fields by keyboard, you have to finalize your changes by pressing `<Enter>` or clicking the *Apply* button.

The controls are grouped as follows:

6.5.1. Time Axis Group

Note that the time axis labeling is precise in terms of relative information only. At offset=0 the absolute time difference between the inputs is close to the shown times but may be off by some tens of picoseconds. Offsets >0 change the placement of the acquisition window and the true time differences are then offset accordingly.

Minimum edit box and spin control

Here the starting value of the displayed time axis can be set. Units are nanoseconds (ns). The minimum is 0. If the entered value exceeds the current time axis maximum, an error message appears. When using the spin control exceeding the current time axis maximum is not possible. Keep in mind the notes on time axis interpretation given in the paragraph *Time Axis Group* above. Also note that the increment upon using the spin buttons grows with the current value. This permits fast changes across the entire range. The mouse wheel can also be used for speedy changes.

Maximum edit box and spin control

Here the end value of the displayed time axis can be set. Units are nanoseconds (ns). If the entered value becomes smaller than the current time axis minimum, an error message appears. When using the spin control, violating the current time axis minimum is automatically prevented. Keep in mind the notes on time axis interpretation given in the paragraph *Time Axis Group* above. Also note that the increment upon using the spin buttons grows with the current value. This permits fast changes across the entire range. The mouse wheel can also be used for speedy changes.

The time axis settings will determine the number of time bins exported via clipboard copy / paste. Only the bins within the displayed time range will be used.

6.5.2. Count Axis Group

Minimum edit box and spin control

Here the starting value of the displayed count axis can be set. Units are counts. If the minimum is set to 0 in logarithmic mode, the resulting minimum displayed is 10^{-1}. If the entered value exceeds the current count axis maximum, an error message appears. When using the spin control, exceeding the current count axis maximum is automatically prevented. Note that the increment upon using the spin buttons grows with the current value. This permits fast changes across the entire range. The mouse wheel can also be used for speedy changes.

Maximum edit box and spin control

Here the upper end of the displayed count axis can be set. Units are counts. If the entered value becomes smaller than the current time axis minimum, an error message appears. When using the spin control, violating the current count axis minimum is prevented automatically. Note that the increment upon using the spin buttons grows with the current value. This permits fast changes across the entire range. The mouse wheel can also be used for speedy changes.

Lin / Log radio buttons

These radio buttons change the scaling of the count axis from linear to logarithmic and vice versa. Despite the fact that in logarithmic scale the display will always show whole powers of ten, the change of the scaling is applied without modifying the minimum / maximum range.

6.6. Trace Mapping Dialog

The PicoHarp 330 software can measure and store histograms in up to 512 memory blocks. Out of these, up to 16 curves can be displayed at the same time. The trace mapping dialog is used to select the curves to display. Tick the individual boxes '*Show*' to display a curve. Select the number of the memory block you wish to map the individual display curve to. The Trace Mapping Dialog also provides some statistics on each curve. These items are:

FWHM	The Full Width at Half Maximum of the curve peak (usually for IRF traces)
Max Count	The count in the highest point of the curve
At Time	The time corresponding to the histogram bin where *Max Count* occurred
Resolution	The time bin width of the curve

Furthermore there are several buttons:

Details	Can be clicked to see more curve information
All	Can be clicked to mark all traces as shown
None	Can be clicked to mark all traces as not shown
0..15	Can be clicked to set the default mapping of trace 0..15 to block 0..15
16..31	Can be clicked to set the mapping of trace 0..15 to block 16..31
32..47	Can be clicked to set the mapping of trace 0..15 to block 32..47
48..63	Can be clicked to set the mapping of trace 0..15 to block 48..63
Clear (trash can)	Can be clicked to delete the contents of individual blocks

The trace mapping dialog can be launched from the corresponding button on the toolbar as well as through the trace color indicator on the control panel.

6.7. General Settings Dialog

Use this dialog to change standard settings of the PicoHarp 330 software. Notably these are: *Prompt overwrite* (warning before overwriting existing data) and *TTTR Marker Settings.* The control connector of the PicoHarp 330 provides TTL inputs for synchronization signals. The markers can be enabled or disabled and recorded at either the rising or falling edge of the corresponding TTL signal. The active edges can be chosen here. All settings will be kept in the Windows registry and will be retrieved at the next program start. They are stored on a per user basis. Note that the retrieved settings may become pitfalls when they are not in agreement with the measurement you are intending to run.

The dialog's controls are grouped as follows:

Display Group

Display Rate /s	Sets the time interval between display updates (from 0.1 to 1s).	
Draw mode	Switches between different curve draw–modes (Lines / Stairs).	
Grid	Checked:	Shows a light grey grid on the curve display.
	Unchecked:	The axes of the curve display are marked with ticks

File Saving Group

Prompt overwrite	Activate this check box if you wish to be prompted before saving data over existing data in a file (recommended).

Warnings Group

Here you can selectively enable / disable individual warnings. Activate the check box for each warning you wish to receive. See section 8.1 for details.

TTTR Marker Settings Group

Here you can set the signal specifications for external marker signals used in TTTR mode (T2 and T3 mode). The settings for the different markers are independent from each other. The following table applies to each of the four markers:

enable	Check this if you want the marker to be included into the TTTR data stream.
rising / falling	Radio buttons to identify the active edge of the signal.

Reference Clock Source Group

Here you can select the PicoHarp's clock source. Normally it runs on its own internal crystal clock. In cases where clock synchronization with other devices is required it is possible to use an external source, e.g., another PicoHarp 330, a GPS disciplined clock, or an atomic clock. The latter are of interest when a more accurate clock is required. In the typical case the clock source will deliver an industry standard 10 MHz clock signal. For more flexibility the PicoHarp 330 also supports external clock rates of 100 and 500 MHz. Note that any change of the clock source requires re-initialization of the PicoHarp 330. Also note that any disconnection from or interruption of the external clock will cause the PicoHarp 330 to fail and report an error. To recover from this error state the device must be re-initialized after the clock source is stable again.

6.8. About PicoHarp 330… Dialog

This dialog provides version information on the PicoHarp 330 software and hardware, the latter only if it is connected and operational. It can be opened via the Help menu or via the toolbar. The button *Request Support* opens a small text viewer window that provides a listing of PicoHarp 330 hardware and software versions that you can copy and paste into your support enquiry. This information is very important for adequate support. Support requests without this information cannot be processed and will be delayed by return questions for this information. If your system is not functional at all, the minimum information you must provide for support is the serial number of your PicoHarp 330. It can be found at the back of the housing. The dialog also provides buttons for links to relevant web pages and a button for checking for and downloading software updates. Note that downloading a software update does not automatically install it. The downloads contain zip files that must be unpacked and installed manually as described in section 3.4. Even though updates are typically bringing improvements and bugfixes, please note that it may not always be advisable to blindly install the latest software. This is the case especially when a user or his/her laboratory have developed custom software that relies on a certain feature of the old version that may have changed. Updates may break such compatibility. Therefore, always carefully read the release notes before installing and/or ensure a safe fallback.

6.9. Title and Comment Editor

You can use this dialog to edit the file title / comment. It can be opened via double–click on the title, via the *File* menu or via the *Print Preview* Toolbar. The text you enter here will be stored in the data file. The first line will be displayed as the file title above the histogram display area. The text you can enter here is limited to 4 lines and 255 characters in total. Upon loading of a data file, the title and comment will also be retrieved. It will also be included in prints, if the corresponding check mark is set in the page setup dialog box available through the *File* menu.

6.10. Print Preview Dialog

Use this Dialog to preview the layout as it would appear when printed. When you choose this command, the main window will be replaced with a print preview window in which one or two pages will be displayed in their printed format. The print preview dialog offers the following options:

> **Print**
>
> Bring up the print dialog box to start a print job.

Next Page

Preview the next printed page.

Prev Page

Preview the previous printed page.

One Page / Two Page

Preview one or two printed pages at a time.

Zoom In

Take a closer look at the printed page.

Zoom Out

Take a larger look at the printed page.

Page Setup

Change the layout of the printed page.

Close

Return from print preview to the editing window.

7.　Problems, Tips & Tricks

7.1.　Basic Pitfalls

The hardware settings in the PicoHarp 330 software are retrieved to the current settings upon loading an existing measurement file. This is a convenient feature most users like because it easily permits repeating a measurement with identical settings. However, the retrieved settings may become pitfalls when they are not in agreement with the measurement you are intending to run. This may be general settings (see section 6.7) as well as control panel settings (see section 6.4). If measurements behave strangely, always check your active settings in both places.

7.2.　PC Performance Issues

The PicoHarp 330 device and its software interface are a complex real–time measurement system requiring appropriate performance both from the host PC and the operating system. This is why a fairly modern CPU and sufficient memory are required, along with a recent USB 3.x (or compatible) host controller. The screen resolution should be at least 800x600. At least a 2 GHz quad core processor, 4 GB of memory and a fast hard disk (preferably SSD) are recommended.

In order to maintain correct interaction between the measurement hardware, the display of histogram curves and user input, the operating system's message passing mechanism is used. It is recommended not to overload the system by running other processes in the background while measuring with the PicoHarp 330. The PC's own occasional network activity should be no problem but running the machine e.g., as a server for other PCs is not recommended. In principle any kind of background activity is allowed. However, should the system become overloaded, photon events may be lost and measurement times and display rates may become irregular.

You can minimize the running PicoHarp 330 software during measurements without issues. This may be of interest for lengthy measurements in integration mode, where one is only interested in accumulating a certain amount of counts without need for permanent monitoring. However, the things you do in the meantime must not overload the CPU.

The "panel meters" showing the current count rates are not meant to be 100 % accurate. They merely serve as an aid for setting up and tuning of the system. Some of them may suffer from system overload. Accordingly, the values shown in the curve details (trace mapping dialog) are also subject to such tolerances.

In TTTR mode, system overload will manifest itself in loss of data and termination of the measurement. Especially the real-time correlator requires significant resources. Here no other heavy load background processes should be allowed to run. The faster the PC, the less these issues will matter. TTTR mode performance (i.e. max. time tagging throughput to disk) may also be improved by using modern solid state disks.

7.3.　Histogram Artefacts

Disturbing histogram ripple is strongly dependent on the quality of the input signals. Try to deliver the best possible signal quality with clean and reasonably steep active edges and without too much ringing. It is recommended to use detector and sync signals of similar amplitude to minimize cross-talk. If a picosecond pulsed laser from PicoQuant's PDL Series is used, attenuating the sync pulses by 10 dB immediately at the laser driver may reduce histogram ripple (set trigger level accordingly). Always use good quality 50 Ω components and coax leads with proper shielding and correct termination. We recommend cables of type RG223 with double braided and silver plated screen. Check your setup for ground loops. Grounding different system components (PC, detector, detector power supply, diode laser driver, etc.) at different points can induce considerable noise in the ground lines. Because signal return paths may share the same ground lines, this noise is copied into the signal lines and causes increased timing jitter and / or histogram ripple. Network cables and mobile phones may also be sources of noise. Use properly impedance matched power splitters (reflection–free T–pads) if signals must be fed to multiple 50 Ω inputs. Never use ordinary BNC tees. All accessories are available from PicoQuant. PMT detectors should be connected through a suitable high speed preamplifier (available from PicoQuant). MCP–PMT detectors should be connected through an amplifier with slightly higher gain (also available from PicoQuant). TTL–SPAD–detectors (e.g. Perkin–Elmer SPCM–AQR and successors by Excelitas) must be connected through an attenuator or attenuating pulse inverter (e.g., PicoQuant SIA 400).

An often overlooked problem in fluorescence lifetime measurements is pile–up and dead time related histogram distortion. It becomes noticeable if the detector count rate exceeds ~5 % of the sync rate. This is why high excitation rates are important. The effect is an inherent problem of high resolution TCSPC and not a fault of the hardware. The PicoHarp 330 is less significantly affected by such issues because of its very short dead time. Nevertheless, the safest way to handle dead time related histogram distortions is to maintain count rates < 5 % of the excitation rate. If high throughput is a primary objective and the detector dead time is not limiting, then this rule of thumb may of course be intentionally ignored. This may bring about the issue of pulse-pile-up because individual detector pulses begin to overlap. Pulse pile-up may also be corrected for in data analysis. Indeed, combined with suitable detectors (e.g., PMA Hybrid from PicoQuant) the PicoHarp 330 will allow fluorescence lifetime measurements with count rates as high as 80 MHz, i.e. as fast as the excitation rate. For further details see the literature given at the end of section 2.4 and the publications on PicoQuant's concept of rapidFLIM.

7.4.   Warm-Up Period

Observe the warm–up period of at about 20 minutes (depending on ambient temperature) before using the PicoHarp 330 for important measurements. You can use this time for set–up and preliminary measurements. The maximum permissible ambient temperature is 35 °C. Make sure that the cooling air can circulate freely and no other hot instrument is directly under the PicoHarp 330.

7.5.   Custom Programming of the PicoHarp 330

A programmer's library (DLL) for custom Windows software development is available to build your own applications e.g., in LabVIEW, Matlab, Python, C/C++, C#, and Pascal (Delphi/Lazarus). A rich set of demo code is provided for an easy start. If you care about performance, consider using a proper compiled high level programming language such as C/C++ or Pascal. Scripted languages like Matlab and Python tend to be very slow. There is also a library version for Linux (x86-64 processor architecture only) which is fully compatible with that for Windows so that applications can easily be ported across the two platforms. The two PH330Lib programming libraries are provided as separate items in the PicoHarp 330 software package. A relatively advanced high-level API package for Python called "snAPI" is also available (separate download from GitHub). It readily provides many real-time analysis methods such as intensity and coincidence time traces, FCS and $g^{(2)}$ correlation.

7.6.   Software Updates

We constantly improve and update the software for our instruments. This includes updates of the configurable hardware (FPGA). Such updates are important as they may affect reliability and interoperability with other products. The software updates are free of charge, unless major new functionality is added. The latest software is available for download at the PicoQuant web site. Alternatively you can click the button 'Check for Updates' in the Help-About dialog available through the main menu or the toolbar. Note that downloading a software update by means of this button does not automatically install it. The downloads contain zip files that must be unpacked and installed manually. Also note that it may not always be advisable to blindly install the latest software. This is the case especially when a user or his/her team have developed custom software that relies on a certain feature that may have changed. Therefore, always carefully read the release notes before installing updates.

7.7.   Support and Bug Reports

The PicoHarp 330 TCSPC system has gone through extensive testing. It builds on over 25 years of experience with several predecessor models and the feedback of hundreds of users. Nevertheless, it is a fairly new product and some bugs may still have gone unnoticed even after rigorous testing. In any case we would like to offer you our support if you experience problems with the system. Do not hesitate to contact PicoQuant in case of difficulties with your PicoHarp 330.

As a first step it is always advisable to study the manual or to press F1 for help. Should you observe errors or bugs caused by the PicoHarp 330 system please try to find a reproducible error situation. Then open *Help - Request Support*. This dialog provides important version information on the PicoHarp 330 software and hardware (the latter only if it is connected and operational). It can be reached via the Help menu or via the toolbar. It provides a small text viewer window with a listing of PicoHarp 330 hardware and software versions that you can copy and paste into your support request. Note that this information is very important for adequate troubleshooting. Support requests without this information cannot be processed and will be delayed by return questions for this information. If your system is not functional at all, the minimum information you must provide for support is the serial number of your PicoHarp 330. It can be found at the back of the housing. There may be other relevant

circumstances, especially other new hardware installed in your PC, so please provide details. You can run *msinfo32* to obtain a listing of your PC configuration and attach the summary file to your error report to www.pi-coquant.com/contact/support. If you cannot access the web form directly, please contact support@picoquant.-com and include the same information*.* Complete information will help us to help you more quickly. When you submit an error report referring to measurement data you have taken, attach the original data file. If the file is too large for email (>5 MB) please provide access to it on a public file server.

If the device must to be sent in for inspection / repair / upgrade, please request an RMA number before shipping the hardware. Unexpected shippings without it will be returned to sender. Take precautions against damage by static discharge, moisture, and mechanical shock under all circumstances in handling, packaging, and shipping. Use the original or equally protective packaging material.

Of course we also appreciate good news: If you have obtained exciting results with one of our systems, please let us know, and where appropriate, please mention the instrument in your publications. At our web–site we maintain a large bibliography of publications related to our instruments. It may serve as a reference for you and other potential users. See http://www.picoquant.com/scientific/references. Please submit your publications for addition to this list.

8. Appendix

8.1. Warnings

When the PicoHarp 330 software is running with functional hardware it continuously collects information about the input signals and the current acquisition settings. If these settings along with the input rates indicate possible errors, the software will indicate this by showing a warning icon.

While the software is running in interactive histogramming mode the warning icon is displayed at the bottom of the main window in the status bar. In the case of TTTR mode, it will appear directly in the respective TTTR mode dialog. The icon can be clicked to display a list of current warnings together with a brief explanation of each warning. Similarly the warnings (if any) will be stored in the data files and can be inspected via the Curve Details dialog, also after re-loading such a file later.

The warnings are to some extent dependent on the current measurement mode. Not all warnings will occur in all measurement modes. Also, count rate limits triggering a specific warning may be different in the various modes. The following table lists the possible warnings in the three measurement modes and gives some explanation as to their possible cause and consequences.

Warning	Histo Mode	T2 Mode	T3 Mode
WARNING_SYNC_RATE_ZERO No pulses are detected at the sync input. In histogramming and T3 mode this is crucial and the measurement will not work without this signal. You may occasionally see this warning if your sync rate is extremely low. If this is intentional you may want to disable this warning. Other warnings may be unreliable under this condition.	√		√
WARNING_SYNC_RATE_VERY_LOW The detected pulse rate at the sync input is below 100 Hz and cannot be determined accurately. If this is intentional you may want to ignore or disable this warning. Other warnings may be unreliable under this condition.	√		√
WARNING_SYNC_RATE_TOO_HIGH The pulse rate at the sync input (after the divider) is higher than 81 MHz. This is close to the limit of sustainable sync front end processing of about 82 MHz. Sync events will be lost above that limit. If you see this warning in T2 mode you may accidentally have connected a fast laser sync. T2 mode is normally intended to be used without a fast sync signal and without a divider.	√	√	√
WARNING_INPT_RATE_ZERO No counts are detected at any of the input channels. In histogramming and T3 mode these are typically the photon event channels and the measurement will yield no event records. You might sporadically see this warning if your detector has a very low dark count rate and is blocked e.g. by a shutter. In that case you may want to disable this warning.	√	√	√

Warning	Histo Mode	T2 Mode	T3 Mode
WARNING_INPT_RATE_TOO_HIGH The overall pulse rate at the input channels is higher than 85 MHz (USB 3.0 connection) or higher than 9 MHz (USB 2.0 connection). This is close to the throughput limit of the present USB connection. The measurement may lead to a FIFO overrun. There are some rare measurement scenarios where this condition is expected and the warning can be ignored or disabled. Examples are short measurements where the FIFO can absorb all data of interest before it runs full.	√	√	√
WARNING_INPT_RATE_RATIO This warning is issued in histogramming and T3 mode when the rate at any input channel is higher than 5% of the sync rate. This is the classic pile-up criterion. There are some measurement scenarios where this condition is expected and the warning can be ignored or disabled. Examples are antibunching measurements or rapidFLIM where pile-up is either tolerated or corrected for during data analysis. One can usually also ignore or disable this warning when the current time bin width is larger than the dead time.	√		√
WARNING_DIVIDER_GREATER_ONE In T2 mode: The sync divider is set larger than 1. This is probably not intended. The sync divider is designed primarily for high sync rates from lasers and requires a periodic pulse train at the sync input. In that case you should use T3 mode. If the signal at the sync input is from a photon detector (coincidence correlation etc.) a divider > 1 will lead to unexpected results. There may be rare measurement scenarios where this condition is intentional and the warning can be ignored or disabled. In histogramming and T3 mode: If the pulse rate at the sync input is below 81 MHz then a divider >1 is normally not needed. The measurement may yield unnecessary jitter if the sync source is not very stable. However, there is one exceptional measurement condition where this warning should be ignored or disabled: If the overall count rate on all channels is very high it may happen that the front end processing is overloaded and starts losing sync events. Consequently some input channel events may appear outside the regular sync period. It is then advisable to increase the sync divider. This requires a periodic sync signal.	√	√	√
WARNING_DIVIDER_TOO_SMALL The pulse rate at the sync input (after the divider) is higher than 81 MHz. This is close to the limit of sustainable front end processing of about 82 MHz. Sync events will be lost above that limit. To avoid this, increase the sync divider. You may also need to increase the divider when the overall count rate on all channels is very high and the front end processing starts losing sync events. Consequently some input channel events may appear outside the regular sync period. In any case recall that using the divider requires a periodic sync signal.	√		√

Warning	Histo Mode	T2 Mode	T3 Mode
WARNING_TIME_SPAN_TOO_SMALL This warning is issued in histogramming and T3 mode when the sync period (1/SyncRate) is longer than the start to stop time span that can be covered by the histogram or by the T3 mode records. You can calculate this time span as follows: Span = Resolution * Length Length is 32768 in T3 mode. In histogramming mode it depends on the histogram length, which is is 65536 in the regular PicoHarp 330 software. It may be chosen differently in custom software. Events outside this span will not be recorded. There are some measurement scenarios where this condition is acceptable or intentional, so the warning can be ignored or disabled.	√		√
WARNING_OFFSET_UNNECESSARY This warning is issued in histogramming and T3 mode when an offset >0 is set even though the sync period (1/SyncRate) can be covered without using an offset (see the coverable time span calcualtion above). The offset may cause events past the end of the coverable time span getting discarded. There are some measurement scenarios where this condition is intentional and the warning can be ignored or disabled.	√		√
WARNING_COUNTS_DROPPED This warning is issued when the front end of the data processing pipeline was not able to process all events that came in. This will occur typically only at very high count rates or during very intense bursts of events.	√	√	√
WARNING_USB20_SPEED_ONLY This warning appears when the PicoHarp 330's USB connection is running only at USB 2.0 speed. For proper performance it should be running at USB 3.0 super speed. Check the cabling and the USB port in use. The same issue is indicated by the USB status LED showing yellow instead of green.	√	√	√

If any of the warnings you receive indicate wrong pulse rates, the cause may be inappropriate input settings, wrong pulse polarities, poor pulse shapes or bad connections. When in doubt, check all signals with an oscilloscope of sufficient bandwidth.

Note that the software can detect only a subset of all possible problematic measurement conditions. It is therefore not safe to assume "all is right" when there is no warning shown. On the other hand, if any of the warnings turns out to be an unnecessary nuisance, e.g., because your specific measurement conditions will inevitably trigger it, you can disable that warning via the General Settings dialog (see section 6.7). Note, however, that disabling the warnings will also apply to which warnings you get to see in the Curve Details dialog when an existing data file is inspected. If you care about this it may be better to keep all warnings enabled.

8.2. Data File Formats

While for many purposes the ASCII export of histograms to files or to the clipboard is sufficient and easy, you also may want to access the PicoHarp 330 data files via custom programs. This section provides only a brief overview on the file format. For details please refer to the online help file available via the help menu.

To overcome certain limitations of various different formats used in the past, PicoQuant now uses a unified file format. It is designed to be future poof in the sense that files created by a current software version stay valid for future software revisions and, moreover, files created by future software versions will most likely still be readable by older software, although they might contain information, that such software can't even "know" about. This is achieved by using a tagged format. Tags identify the data to follow, and give the type, length and even meta information. The exact location of an individual item in the file is then irrelevant. Version robustness is granted as long as version-breaking changes to the semantics of a given field are implemented by a tag with a new identifier rather than expanding the range or interpretation of the old one. The list of tags (identifiers) and their interpretation rules can be kept in a tag dictionary. With this as a precondition, the software only has to show tolerance on missing non-mandatory (i.e. optional) content.

The new format definition unifies PicoQuant's existing file formats which individually evolved over many years. The resulting new TTTR file format with the extension `*.ptu` will be used for all current and future TCSPC products supporting TTTR mode and enriches them with powerful new features. Similarly, a tagged file format with the extension `*.phu` covers the histogram data formats of our current and future TCSPC products.

To support understanding of the format and implementation of custom software accessing these files, a set of demos is provided in the subfolder `\Filedemo` in your chosen software installation folder. If you need to evaluate more header items than the demos do, please refer to the PicoHarp 330 online help file available via the help menu. A file format related HTML help file is also provided in the file demo folder. It contains a list of tag types and a tag dictionary that explains the individual items. Note that the dictionary contains more items than the PicoHarp 330 software actually uses. It is recommended to go by a specific file, have one of the demos read it and then look at the list of header items you get. You can also use the PicoQuant File Info shell extension that will be installed by the PicoHarp 330 software setup to inspect individual header items of a `*.ptu` or `*.phu` file. Just right-click on the file in Windows explorer and select *Properties*.

Despite the intended version tolerance of the tagged format, for consistency and safe version checking the PicoHarp 330 data files still carry a format version number, which is now called *content version* and currently has the string value "1.0". In order to identify a PicoHarp 330 data file as a file created by and to be used by the native PicoHarp 330 software there is a tag *assured content* which begins with the string "PicoHarp 330". There is also a pair of tags for creator name and creator version that identify the creating software. Programmers of custom software writing such files MUST USE THEIR OWN CREATOR NAME.

Note that despite our best efforts towards compatibility and version tolerance, file formats in future software releases are subject to change without notice.

8.2.1. Interactive Mode File Format

The standard PicoHarp 330 histogram data files created by the PicoHarp 330 software in interactive histogramming mode (`*.phu`) are tagged binary files which contain both the setup parameters and the actual histogram data. The latter can be present multiple times, i.e. multiple measurements can be stored in one file. Relevant settings are stored for each measurement separately. In order to identify a PicoHarp 330 data file as a file created by and to be used by the native PicoHarp 330 software, a program reading in these files can read the tag *assured content* which begins with the string "PicoHarp 330". However, a piece of software aiming solely at retrieving the histogram data content can (and should) be tolerant about this tag and go for the pure histogram data. This tolerance will ensure compatibility for the future. Indeed, the demos in the subfolder `\Filedemo` in your chosen installation directory are following this tolerant approach. For more information on individual file tags and their content, please consult the online help file available via the help menu.

8.2.2. TTTR Mode File Format

PicoHarp 330 data files from T2 and T3 Mode (`*.ptu`) created by the PicoHarp 330 software are tagged binary files which contain both the setup parameters and the actual event data. There can be only one measurement per file. The setup data in the file header is similar to that in standard interactive mode files. In order to identify a PicoHarp 330 data file as a file created by and to be used by the native PicoHarp 330 software, a program processing these files can read the tag *assured content* which begins with the string "PicoHarp 330". However, a

piece of software aiming solely at retrieving the event record data content can (and should) be tolerant about this tag and go for the pure event record data. This tolerance will ensure compatibility for the future. The demos in the subfolder `\Filedemo` in your chosen installation directory are following this tolerant approach. For more information on individual file tags and their content, please consult the online help file available via the help menu. It is worth noting that the actual TTTR record data following the file header corresponds directly to the raw data obtained with custom programs using the PicoHarp 330 programming library.

8.3.   Hardware Technical Data

8.3.1.   Specifications

All information given here is reliable to our best knowledge. However, no responsibility is assumed for possible inaccuracies or omissions. Specifications and external appearance are subject to change without notice.

Input Channels and Sync

Number of Input Channels	Sync + 1, 2, 3, or 4
Trigger principle	edge trigger (ETR) or constant fraction discriminator (CFD) software selectable
ETR Input voltage operating range (pulse peak) optimum:	−1500 … 1500 mV abs. Amplitude 200..500 mV
ETR level adjust	−1500 … 1500 mV Resolution 1 mV
ETR trigger edge	rising or falling, software selectable
CFD Input voltage operating range (pulse peak) optimum:	−1500 … 0 mV -500..-200 mV
CFD discriminator level adjust	−1500 … 0 mV Resolution 1 mV
CFD zero cross level adjust	−100 … 0 mV Resolution 1 mV
Input impedance	50 Ω
Input voltage max. range (damage level)	−2000 … 3000 mV
Input pulse width	≥ 0.25 ns
Input pulse rise ∕ fall time	≤ 20 ns
Marker and control inputs	standard TTL, > 2 kΩ DC load

Time to Digital Converters

Base resolution (min. time bin width)		1 ps
Single measurement timing uncertainty (rms)*		typ. 2 ps
Start-stop timing uncertainty (rms)*		typ. 3 ps
	PicoHarp 330 2P:	$+ 10^{-8} \cdot \Delta t$
	PicoHarp 330 4P:	$+ 10^{-10} \cdot \Delta t$
Differential non–linearity	PicoHarp 330 2P:	< 10% pp, < 1% rms
	PicoHarp 330 4P:	< 6% pp, < 0.9% rms
Dead time		< 680 ps
Peak count rate		$1.47 \cdot 10^{9}$ cps sustainable for bursts of 1000 events
Max. sustained count rate		$82 \cdot 10^{6}$ cps
Max. sync rate without divider		81 MHz
Max. sync rate with divider		640 MHz
Adjustable delay range for each input delay resolution = base resolution		± 100 ns

* In order to determine the timing uncertainty or "jitter" it is necessary to repeatedly measure a time difference and to calculate the standard deviation (rms error) of these measurements. This is done by splitting an electrical signal from a pulse generator and feeding the

two signals each to a separate input channel. The differences of the measured pulse arrival times are then calculated along with the corresponding standard deviation. This latter value is the rms jitter or timing uncertainty of start-stop measurements. However, calculating such time differences requires two measurements. These both have a small random jitter and a term that is dependent on the time difference between the two signals because over longer spans the crystal clock jitter becomes noticeable. For short term measurements on a sub-microsecond scale, as in most TCSPC applications, the latter is typically negligible. The timing precision of a single measurement, according to error propagation laws, is obtained by dividing the start-stop timing uncertainty by $\sqrt{2}$. We specify this single measurement rms error here for comparison with other products.

Histogrammer

Maximum number of time bins	65,536 via GUI 524,288 via DLL
Full scale time range (depending on chosen resolution)	65.5 ns … 549.7 ms via GUI 65.5 ns … 4.398 s via DLL
Count depth per time bin	4,294,967,296 (32 bit)
Acquisition time	1 ms … 100 h
Sustained throughput to PC memory (total for all channels)	typ. 85×10^6 events/sec (depending on host PC configuration and performance)

TTTR Engine

T2 mode resolution	1 ps
T3 mode resolution (settable)	1 ps … 4.19 µs
T3 Mode number of time bins	32768
FIFO buffer depth (records)	268,435,456
Sustained throughput over USB (sum of all channels)	typ. 85×10^6 events/sec (depending on host PC configuration and performance)
Acquisition time	1 ms … 100 h
External marker inputs (TTL)	4
External marker input pulse duration	> 50 ns
External marker input pulse spacing	> 50 ns
External marker rise/fall time	< 50 ns
External Marker timing error in T2 mode	< 100 ns
External Marker timing error in T3 mode	< Tsync + 100 ns

External Reference Clock

Reference input	10 MHz, 100 MHz, 500 MHz 200 … 1 500 mV p.p. 50 Ω; AC coupled
Reference output	10 MHz 1000 mV p.p. 50 Ω; AC coupled

Trigger Output

Baseline level	0V typ.
Pulse amplitude	-0.6V typ.
Pulse duration	10 ns typ.
Load/Termination	50 Ω
Programmable period range	0.1 µs ... 1.678 s

Operating Environment

Environment restrictions	Indoor use only

Ambient temperature	≤ 35°C
Relative humidity	< 80% non-condensing
Operation altitude	max. 2000 m above sea level
Warm–up period to meet specs	20 min
Recommended PC specifications	
CPU	≥ 4 cores, ≥ 2 GHz
RAM	≥ 4 GB
USB minimum required	USB 2.0
USB required to meet specs	USB 3.0 or compatible higher
Operating system	Windows 10 or 11 x86-64 or Linux x86-64 with Wine (see section 8.4)

Power Supply

Line voltage	100...240V AC 50...60 Hz
Power consumption	≤ 25 W
Fuse	2 x T1.6A/250V

Dimensions

Size incl. feet and handles	305 × 240 × 95 mm
Weight	2.5 kg

Retraction of Discarded Devices

Waste electrical products must not be disposed of with household waste.

This equipment should be taken to your local recycling center for safe treatment.

WEEE–Reg.–Nr. DE 96457402

8.3.2. Connectors

The inputs for the photon detector signals and the sync signal are SMA connectors located on the front panel of the PicoHarp 330. They are labeled **SYNC, CH1, CH2, …** (model dependent up to 4). The inputs are terminated with 50 Ohms internally. Use quality 50 Ohms coax cables with appropriate connectors. For interfacing to BNC connectors use standard adapters. Carefully screw on the SMA connectors for sync and detector(s) until they are hand-tight. Do not use wrenches. Note that PMT detectors operate with high voltages that may discharge through the signal cable. Make sure such detectors are switched off and fully discharged before connecting them.

Apart from the SMA connectors for the input signals the PicoHarp 330 has further connectors for USB, clock synchronization, trigger output, and experiment control.

The **USB connector** (blue) is a standard USB 3.0 type B socket located at the front panel. It is used to connect to the host PC via a qualified USB 3.0 cable. Observe the notes in section 3.5 for making this connection.

The **control connector** is a 25-pin female sub-D connector labeled 'CTRL'. It is located at the back of the housing. Note that future firmware/software is going to allow reconfiguration of some of these pin assignments. The following figure shows the pin layout and the table below contains the connector's default pin assignments.

male plug – front (pin) view

female plug – front (hole) view
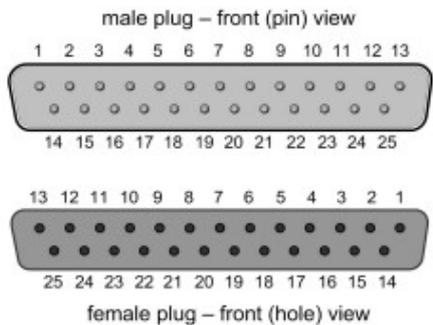
CTRL Connector – Pin numbering scheme

CTRL Connector – Default Pin Assignments			
Pin#	Name	I/O	Purpose/Description
1	GPIO 0	TTL in	marker 1 input
2	GPIO 1	TTL in	marker 2 input
3	GPIO 2	TTL in	marker 3 input
4	GPIO 3	TTL in	marker 4 input
5	GPIO 4		reserved
6	GPIO 5		reserved
7	GPIO 6		reserved
8	GPIO 7		reserved
9	GPIO 8		reserved
10	GPIO 9		reserved
11	GPIO 10		reserved
12	GPIO 11		reserved
13	GPIO 12		reserved
14	GND	GND	Ground (0V)
15	GPIO 13		reserved
16	GPIO 14		reserved
17	GPIO 15		reserved
18	GND	GND	ground (0V)
19	C1	TTL in	start measurement (requires dedicated software)
20	C2	TTL in	stop measurement (requires dedicated software)
21	GND	GND	ground (0V)
22	MACT	TTL out	high when measurement running
23	GND	GND	ground (0V)
24	D3V3	DC out	+3.3V / ≤ 350 mA supply for external hardware add-ons
25	GND	GND	ground (0V)

The specifications for TTL signals can be found at http://wikipedia.org/wiki/Transistor-transistor_logic. Note that for the TTL inputs described here a maximum high level of 5V is permitted.

For further details regarding signal specifications, notably the trigger output (voltages, frequencies pulse widths etc.) see section 8.3.1.

Pins 1, 2, 3 and 4 accept 3.3V TTL compatible synchronization signals that will be recorded as markers in TTTR mode. The pins are internally pulled down, so that they are inactive when left unconnected. The active edge is chosen in the software settings dialog. Rise/fall times must be 50 ns or faster. Both high and low state must be at least 50 ns long. The clock period may therefore (in principle) be as short as about 100 ns but data bus throughput constraints will apply. Each marker creates an additional TTTR record, so that one must ensure not to swamp the data stream with too many marker records. When bandwidth gets tight, markers take precedence over photon records, so that excess marker traffic can suppress photon records. In fast imaging applications it is therefore recommended not to use a pixel clock but a line clock only. Because each photon has a time tag, it is usually not necessary to use an additional pixel clock. For more information on how to use the marker inputs see section 5.3.5.

Pins 19 and 20 can be used to implement hardware triggered measurements. Note that this requires custom software (see the DLL manual and related demos).

Pin 22 is a TTL output that goes high when a measurement is running.

Pin 24 provides a 3.3 V DC supply voltage that external electronics can use. Under no circumstances must this line be shorted to ground or loaded in excess of the specified maximum current.

Pins 14, 18, 21, 23, and 25 are the common ground for the TTL signals and the DC supply line. They are also connected to the housing.

Make sure not to confuse the control connector with a connector for other equipment that would physically fit but might lead to mutual damage. Appropriate cables for typical applications of the control port are available from PicoQuant.

At the back of the housing the PicoHarp 330 there is a standard **connector for power supply** that includes two fuses. The fuses are the device's only user serviceable parts. Please find the required fuse current ratings on the product label or in section 8.3.1. Do NOT replace the fuses with one of higher current rating than specified. Use slow blowing fuses of the specified rating only. If the fuse blows repeatedly stop further experiments and contact support.

At the back there are also another four SMA connectors labeled "TRG OUT", "REF IN", and "REF OUT".

TRG OUT is a programmable trigger output for pulsed light sources. Note that this is under software control and must be used with greatest care when potentially dangerous laser sources are triggered from it. The trigger rate is programmable from 0.1 µs to 1.678 s (0.596 Hz to 10 MHz). The pulse width is about 10 ns, the base line level is near 0 V and the active level (pulse peak) is approximately -0.7 V. When a light source is triggered from this signal the same signal must typically be fed to the sync input. This requires proper impedance matching e.g., by means of a power splitter (reflection-free T-pad).

REF IN accepts an industry standard 10 MHz clock reference, e.g., from a frequency normal, an atomic clock, or another PicoHarp 330. When this clock source is selected (via software) then the PicoHarp 330 will lock its internal PLL to this signal. Note that it must not be disconnected once it is being used as the active clock source. For extended flexibility the PicoHarp 330 also accepts 100 and 500 MHz signals as a reference clock. Which clock frequency is to be used can be configured by software.

REF OUT provides an industry standard 10 MHz clock output phase locked to the PicoHarp 330's internal clock, e.g., for the REF IN of another PicoHarp 330. Once the other PicoHarp 330 has locked its internal PLL to this signal the clocks of the two devices will not drift apart.

8.3.3. Indicators

The PicoHarp 330 shows some status information by means of four LEDs on the front panel. The meaning of these indicators, from top to bottom, is as follows:

ACTIVE

off   = measurement not active
green   = measurement active
yellow   = measurement active, warning: events dropped

The on/off state reflects the state of pin 22 of the CTRL connector (high when measurement running).

LOCK

green   = PLL locked
yellow   = reconfiguring, PLL not yet locked
red   = PLL not locked

This is particularly relevant when an external clock is used. The LED will then indicate if the PicoHarp 330 has actually locked itself to that frequency.

ERROR

off   = device not yet initialized
red   = error
yellow   = warning, check software warnings for details
green   = no error

USB

off   = not yet connected
yellow   = USB 2.0 connection, warning: poor throughput
green   = USB 3.0 connection
red   = disconnected

The USB LED showing green or yellow indirectly indicates that the device driver is loaded.

Please note that the LEDs can only give a very crude indication of errors or warnings. The indicated errors or warnings are also monitored by the software, which gives much more tangible information on the possible issues. The most likely warning you may see indicated by yellow light is when the input rate is too high and events are dropped. However, in some cases the LEDs may actually light up only briefly. Looking at the status information given by the software should therefore be preferred.

8.4. Using the Software under Linux

The PicoHarp 330 software can also be used under Linux (x86-64 platform only). This requires that Wine is installed (see https://www.winehq.org). You can run the regular software setup as explained in section 3.4. Instead of installing a device driver, running under Linux with Wine requires that you have Libusb 1.0 installed (see https://libusb.info). The installer setup.exe will attempt do detect Wine and then automatically skip the Windows device driver installation. NOTE, however, that on newer Wine installations this detection may fail. It is then important NOT to install the device driver. You can de-select it at one of the initial installer wizard pages or at the moment the driver installation wizard appears.

We have successfully tested these configurations:

- Wine 7.0 and Libusb 1.0.23 on Ubuntu 20.04 LTS.
- Wine 8.0 and Libusb 1.0.25 on Ubuntu 22.04 LTS.
- Wine 9.16 and Libusb 1.0.27 on Ubuntu 24.04 LTS   See NOTE above.

In all cases we used the package `winehq-stable` which is typically more mature than the version provided by the Linux distribution.

Access Permissions

For device access through libusb suitable permissions for the device must be granted to the normal user, otherwise only the super-user `root` will have access. Recent Linux distributions use udev to handle this. For automated setting of the device access permissions with udev you can add an entry to the set of rules files that are contained in `/etc/udev/rules.d`. Udev processes these files in alphabetical order. The default rule files usually carry names starting with a number. Don't change these files as they could be overwritten when you upgrade your system. Instead, put your custom rule for the PicoHarp 330 in a separate file. The typical content of this file should be:

```
ATTR{idVendor}=="0d0e", ATTR{idProduct}=="0015", MODE="666"
```

A suitable rules file `PicoHarp330.rules` is provided in the folder `udev` under the distribution pack folder `PicoHarp 330 v.X.X.X.X\installation\Linux`. You can simply copy it to the `/etc/udev/rules.d` folder. The install script in the same folder does just this. Note that this requires root permissions. As a normal user you must run it preceded with `sudo`. After that you need to disconnect and reconnect the device to get access.

If you have issues obtaining permissions recall that the name of the rules file is important. Each time a device is detected by the udev system, the files are read in alphabetical order until a match is found. Different Linux distributions may use different rule file names for various categories. If there happen to be later rules that are more general (applying to a whole class of devices) they may override your custom rule and the desired access rights. It is therefore important that you use a rules file named such that it gets evaluated <u>after</u> the general case. The default naming `PicoHarp330.rules` most likely ensures this but if you see access problems you may want to check.

Note that the setting `MODE="666"` is quite permissive for all users. If you prefer tighter security regarding device access please study the documentation of udev and/or the recommendations of your distribution for handling USB device access, e.g. employing user classes with suitable access rights.

Wine Limitations

Running the software under Wine is an experimental feature with limited support. A known issue is the current implementation of Wine's viewer for chm help files. You may observe failure or some glitches when using the online help facility.