# FPGA implementation of a 32x32 autocorrelator array for analysis of fast image series

**Jan Buchholz,[1] Jan Wolfgang Krieger,[1] Gábor Mocsár,[2] Balázs Kreith,[2] Edoardo Charbon,[3] György Vámosi,[2] Udo Kebschull[4] and Jörg Langowski[1,*]**

[1] *German Cancer Research Center (DKFZ), Biophysics of Macromolecules (B040), Im Neuenheimer Feld 580, D-69120 Heidelberg, Germany*
[2] *University of Debrecen, Medical and Health Science Center, Faculty of Medicine, Department of Biophysics and Cell Biology, H-4032 Debrecen, Nagyerdei krt. 98, Hungary*
[3] *Technische Universiteit Delft, Mekelweg 4, 2628 CD Delft, The Netherlands*
[4] *Goethe-Universität Frankfurt, Senckenberganlage 31, D-60325 Frankfurt, Germany*
[*] *jl@dkfz.de*

*http://www.dkfz.de/Macromol*

**Abstract:** With the evolving technology in CMOS integration, new classes of 2D-imaging detectors have recently become available. In particular, single photon avalanche diode (SPAD) arrays allow detection of single photons at high acquisition rates ($\geq$ 100 kfps), which is about two orders of magnitude higher than with currently available cameras. Here we demonstrate the use of a SPAD array for imaging fluorescence correlation spectroscopy (imFCS), a tool to create 2D maps of the dynamics of fluorescent molecules inside living cells. Time-dependent fluorescence fluctuations, due to fluorophores entering and leaving the observed pixels, are evaluated by means of autocorrelation analysis. The multi-$\tau$ correlation algorithm is an appropriate choice, as it does not rely on the full data set to be held in memory. Thus, this algorithm can be efficiently implemented in custom logic. We describe a new implementation for massively parallel multi-$\tau$ correlation hardware. Our current implementation can calculate 1024 correlation functions at a resolution of 10 $\mu$s in real-time and therefore correlate real-time image streams from high speed single photon cameras with thousands of pixels.



**OCIS codes:** (040.0040) Detectors; (040.1240) Arrays; (040.1345) Avalanche photodiodes (APDs); (040.1490) Cameras; (180.2520) Fluorescence microscopy; (180.6900) Three-dimensional microscopy; (300.6280) Spectroscopy, fluorescence and luminescence; (100.4550) Correlators.

## References and links

1. D. Magde, E. L. Elson, and W. W. Webb, "Fluorescence correlation spectroscopy i: conceptual basis and theory," Biopolymers **13**, 1–27 (1974).
2. D. Magde, E. L. Elson, and W. W. Webb, "Fluorescence correlation spectroscopy. ii. an experimental realization," Biopolymers **13**, 29–61 (1974).
3. O. Krichevsky and G. Bonnet, "Fluorescence correlation spectroscopy: the technique and its applications," Rep. Prog. Phys. **65**, 251–297 (2002).

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

4. M. Engels, B. Hoppe, H. Meuth, and R. Peters, "A single chip 200 MHz digital correlation system for laser spectroscopy with 512 correlation channels," in "ISCAS'99. Proceedings of the 1999 IEEE International Symposium on Circuits and Systems, 1999," , vol. 5 (IEEE, 1999), vol. 5, pp. 160–163.
5. B. Hoppe, H. Meuth, M. Engels, and R. Peters, "Design of digital correlation systems for low-intensity precision photon spectroscopic measurements," in "IEEE Proceedings Circuits, Devices and Systems," , vol. 148 (IET, 2001), vol. 148, pp. 267–271.
6. M. Engels, B. Hoppe, H. Meuth, and R. Peters, "Fast digital photon correlation system with high dynamic range," in "Proceedings of the 13th Annual IEEE International ASIC/SOC Conference, 2000," (IEEE, 2000), pp. 18–22.
7. M. Wahl, I. Gregor, M. Patting and J. Enderlein, "Fast calculation of fluorescence correlation data with asynchronous time-correlated single-photon counting," Opt. Express **11**, 3583–3591 (2003).
8. T. Laurence, S. Fore, and T. Huser, "A fast, flexible algorithm for calculating correlations in fluorescence correlation spectroscopy," Opt. Lett. **31**, 829–31 (2006).
9. E. Schaub, "F2cor: fast 2-stage correlation algorithm for FCS and DLS," Opt. Express **20**, 2184–2195 (2012).
10. D. Magatti and F. Ferri, "Fast multi-tau real-time software correlator for dynamic light scattering," Appl. Opt. **40**, 4011–4021 (2001).
11. D. Magatti and F. Ferri, "25 ns software correlator for photon and fluorescence correlation spectroscopy," Rev. Sci. Instrum. **74**, 1135–1144 (2003).
12. M. Culbertson and D. Burden, "A distributed algorithm for multi-tau autocorrelation," Rev. Sci. Instrum. **78**, 044102 (2007).
13. B. Tieman, S. Narayanan, A. Sandy, and M. Sikorski, "Mpicorrelator: a parallel code for performing time correlations," Nucl. Inst. Meth. A **649**, 240–242 (2011).
14. C. Jakob, A. Schwarzbacher, B. Hoppe, and R. Peters, "The development of a digital multichannel correlator system for light scattering experiments," in "Irish Signals and Systems Conference, 2006. IET," (IET, 2006), pp. 99–103.
15. C. Jakob, A. T. Schwarzbacher, B. Hoppe, and R. Peters, "A FPGA optimised digital real-time mutichannel correlator architecture," in "10th Euromicro Conference on Digital System Design Architectures, Methods and Tools, 2007. DSD 2007," (IEEE, 2007).
16. C. Jakob, A. Schwarzbacher, B. Hoppe, and R. Peters, "A multichannel digital real-time correlator as single FPGA implementation," in "15th International Conference on Digital Signal Processing, 2007," (2007), pp. 276–279.
17. Y. Yang, J. Shen, W. Liu, and Y. Cheng, "Digital real-time correlator implemented by field programmable gate array," in "CISP'08. Congress on Image and Signal Processing, 2008," , vol. 1 (IEEE, 2008), vol. 1, pp. 149–151.
18. W. Liu, J. Shen, and X. Sun, "Design of multiple-tau photon correlation system implemented by FPGA," in "ICESS'08. International Conference on Embedded Software and Systems, 2008," (IEEE, 2008), pp. 410–414.
19. G. Mocsar, B. Kreith, J. Buchholz, J. W. Krieger, J. Langowski, and G. Vamosi, "Note: multiplexed multiple-tau auto- and cross-correlators on a single field programmable gate array," Rev. Sci. Instrum. **83**, 046101 (2012).
20. M. Burkhardt and P. Schwille, "Electron multiplying ccd based detection for spatially resolved fluorescence correlation spectroscopy," Opt. Express **14**, 5013–5020 (2006).
21. R. A. Colyer, G. Scalia, I. Rech, A. Gulinatti, M. Ghioni, S. Cova, S. Weiss, and X. Michalet, "High-throughput FCS using an LCOS spatial light modulator and an $8 \times 1$ SPAD array," Biomed. Opt. Express **1**, 1408–1431 (2010).
22. R. Colyer, G. Scalia, F. Villa, F. Guerrieri, S. Tisa, F. Zappa, S. Cova, S. Weiss, and X. Michalet, "Ultra high-throughput single molecule spectroscopy with a 1024 pixel SPAD," in "Proc. SPIE," **7905**, 790503–1 (2011).
23. G. Heuvelman, F. Erdel, M. Wachsmuth, and K. Rippe, "Analysis of protein mobilities and interactions in living cells by multifocal fluorescence fluctuation microscopy," Eur. Biophys. J. **38**, 813–828 (2009).
24. F. Bestvater, Z. Seghiri, M. S. Kang, N. Gröner, J. Y. Lee, I. Kang-Bin, and M. Wachsmuth, "EMCCD-based spectrally resolved fluorescence correlation spectroscopy," Opt. Express **18**, 23818–23828 (2010).
25. D. J. Needleman, Y. Xu, and T. J. Mitchison, "Pin-hole array correlation imaging: highly parallel fluorescence correlation spectroscopy," Biophys. J. **96**, 5050–5059 (2009).
26. B. Kannan, L. Guo, T. Sudhaharan, S. Ahmed, I. Maruyama, and T. Wohland, "Spatially resolved total internal reflection fluorescence correlation microscopy using an electron multiplying charge-coupled device camera," Anal. Chem. **79**, 4463–4470 (2007).
27. T. Wohland, X. Shi, J. Sankaran, and E. H. K. Stelzer, "Single plane illumination fluorescence correlation spectroscopy (SPIM-FCS) probes inhomogeneous three-dimensional environments," Opt. Express **10**, 10627–10641 (2010).
28. J. Capoulade, M. Wachsmuth, L. Hufnagel, and M. Knop, "Quantitative fluorescence imaging of protein diffusion and interaction in living cells," Nat. Biotechnol. **29**, 835–839 (2011).
29. L. Carrara, C. Niclass, N. Scheidegger, H. Shea, and E. Charbon, "A gamma, x-ray and high energy proton radiationtolerant CMOS image sensor for space applications," in "ISSCC, IEEE International Solid-State Circuits Conference," (2009), pp. 40–41.
30. M. Gösch, A. Serov, T. Anhut, T. Lasser, A. Rochas, P. Besse, R. Popovic, H. Blom, and R. Rigler, "Parallel single molecule detection with a fully integrated single-photon $2 \times 2$ CMOS detector array," J. Biomed. Opt. **9**,

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

913 (2004).
31. R. Colyer, G. Scalia, T. Kim, I. Rech, D. Resnati, S. Marangoni, M. Ghioni, S. Cova, S. Weiss, and X. Michalet, "High-throughput multispot single-molecule spectroscopy," in "Proceedings-Society of Photo-Optical Instrumentation Engineers," , vol. 7571 (NIH Public Access, 2010), vol. 7571, p. 75710G.
32. C. Veerappan, J. A. Richardson, R. J. Walker, D.-U. Li, M. W. Fishburn, Y. Maruyama, D. Stoppa, F. Borghetti, M. Gersbach, R. K. Henderson, and E. Charbon, "A 160x128 single-photon image sensor with on-pixel 55ps 10b time-to-digital converter." in "ISSCC, IEEE International Solid-State Circuits Conference," (IEEE, 2011), pp. 312–314.
33. C. Niclass, M. Sergio, and E. Charbon, "A single photon avalanche diode array fabricated in 0.35-$\mu$m CMOS and based on an event-driven readout for TCSPC experiments," in "Proc. SPIE," **6372**, 63720S (2006).
34. K. Schätzel, "Noise on photon correlation data: I. autocorrelation functions," Quantum Opt. **2**, 287–305 (1990).
35. K. Schätzel, "New concepts in correlator design," Inst. Phys. Conf. Ser. **77**, 175–184 (1985).
36. Z. Kojro, A. Riede, M. Schubert, and W. Grill, "Systematic and statistical errors in correlation estimators obtained from various digital correlators," Rev. Sci. Instrum. **70**, 4487–4496 (1999).
37. J. Sankaran, X. Shi, L. Ho, E. Stelzer, and T. Wohland, "ImFCS: a software for imaging FCS data analysis and visualization," Opt. Express **18**, 25468–25481 (2010).
38. The diffusion coefficient was $D = 20\,\mu\mathrm{m}^2/\mathrm{s}$ (corresponding to an intermediately sized protein in water), the simulation timestep of the random walk, as well as the minimum lag time were $\Delta t_{\mathrm{sim}} = \tau_{\mathrm{min}} = 1\,\mu\mathrm{s}$. There were around 1.2 particles in the effective measurement volume $V_{\mathrm{eff}} \approx 0.4\,\mu\mathrm{m}^3$ on average.
39. T. Wocjan, J. Krieger, O. Krichevsky, and J. Langowski, "Dynamics of a fluorophore attached to superhelical DNA: FCS experiments simulated by brownian dynamics," Phys. Chem. Chem. Phys. **11**, 10671–10681 (2009).
40. C. Niclass, C. Favi, T. Kluter, M. Gersbach, and E. Charbon, "A 128 × 128 single-photon imager with on-chip column-level 10b time-to-digital converter array capable of 97ps resolution," in "ISSCC, IEEE International Solid-State Circuits Conference," (IEEE, 2008), pp. 44–594.
41. K. Greger, J. Swoger, and E. H. K. Stelzer, "Basic building units and properties of a fluorescence single plane illumination microscope," Rev. Sci. Instrum. **78**, 023705 (2007).
42. Joachim Wuttke: lmfit - a C/C++ routine for Levenberg-Marquardt minimization with wrapper for least-squares curve fitting, based on work by B. S. Garbow, K. E. Hillstrom, J. J. Moré, and S. Moshier. Version 3.2, retrieved on 2011-08-31 from http://www.messen-und-deuten.de/lmfit/.
43. QuickFit 3.0 can be downloaded free of charge from http://www.dkfz.de/Macromol/quickfit/. In addition to the fitting capabilities, it also contains software implementations of the correlators described in here.
44. S. T. Hess and W. W. Webb, "Focal volume optics and experimental artifacts in confocal fluorescence correlation spectroscopy," Biophys. J. **83**, 2300–2317 (2002).

## 1. Introduction

Fluorescence correlation spectroscopy (FCS) [1,2] is a powerful experimental technique for measuring the dynamics of fluorescently labeled molecules in solution and also inside living cells. It allows one to determine the particle number, the diffusion coefficient, flow speeds and also photophysical and chemical reaction rates (for an overview, see Ref. [3]). In FCS the time trace of the fluorescence intensity fluctuations $I(t)$ inside a small observation volume (usually around $10^{-15}\mathrm{l} = 1\,\mu\mathrm{m}^3$) is monitored. The fluctuations originate from particles entering and leaving the focus, or transitions between states having different quantum yields. Faster dynamics of the fluorescing particles also lead to faster fluctuations, which can be quantified by means of a temporal first-order autocorrelation function (ACF):

$$g(\tau) = \frac{\langle I(t)\cdot I(t+\tau)\rangle_t}{\langle I(t)\rangle_t^2}, \quad \langle I(t)\rangle_t := \lim_{\bar{T}\to\infty}\frac{1}{\bar{T}}\int_0^{\bar{T}} I(t)\,\mathrm{d}t \tag{1}$$

The ACF usually contains features that are spread over several orders of magnitude in time (nanoseconds to seconds), with $\bar{T}$ being the runtime of the entire measurement.

The standard FCS setup uses a confocal microscope in combination with single-photon sensitive detectors to acquire the fluorescence time trace $I(t)$ from one focal volume. Then the data is fed into a "correlator" (hardware or software component), which estimates the ACF over a certain dynamic range.

Several hard- and software implementations for correlators (especially for the application in confocal FCS and dynamic light scattering) have been proposed. On the one hand, there are

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

custom hardware based approaches (e.g. [4–6]). On the other hand, highly optimized software algorithms have been invented which mostly rely on photon arrival times [7–9], or commercial event counter cards [10,11]. Parallelized implementations of the multi-$\tau$ algorithm are available also [12,13]. More and more reconfigurable logic chip (FPGA – field programmable gate array) based approaches have been developed [14–19] lately. Most of them can only deal with a limited number of up to 32 input channels. Particularly, the software implementations achieve good (near-realtime) performance only for a very limited number of input channels, if implemented on a standard PC.

In recent years, the availability of fast cameras has triggered the development of different spacially resolved modalities for FCS: these modalities are based on single-spot confocal microscopy [20], multi-spot confocal microscopy [21, 22], line-confocal microscopy [23, 24], spinning-disk microscopy [25], total internal reflection microscopy [26] and selective plane illumination microscopy [27,28]. All these techniques map the diffusion coefficients and other dynamic properties by calculating an ACF for each of potentially many pixels in software. Also the commercially available fast cameras are limited in frame rate to about 1 kfps if a substantial number of pixels needs to be measured, although they can achieve frame rates up to 25 kfps for single lines (see e.g. [28]).

Here we describe a hardware-based correlation system that can process the data stream from all pixels of an imaging device in real time. We extend the idea of hardware reuse, as presented in Refs. [14, 19], for imaging FCS applications with hundreds of input signals. Our FPGA-based hardware correlator system can calculate 1024 autocorrelation functions in parallel and in real time. The dynamic range of the ACFs is $\tau_{\mathrm{min}} \ldots \tau_{\mathrm{max}} = 10\,\mu\mathrm{s} \ldots 1\,\mathrm{s}$. The design could easily be adapted to different kinds of image sensors such as single photon avalanche diode (SPAD) arrays, customized scientific complementary metal oxide semiconductor (sCMOS) or electron multiplying charge-coupled device (EMCCD) cameras.

To overcome the limited temporal resolution of currently available cameras, we use the $32 \times 32$ pixel SPAD array detector *Radhard2* as image sensor, which can be read at frame rates of 100 kfps and above (for a detailed description of this sensor, please refer to section 2 and Ref. [29]). Smaller SPAD arrays have already been used for parallelized FCS on multi-focus confocal microscopes [30,31]. Also a subregion of a SPAD array with the same size, but with very different pixel architecture has been used for FCS [22]. In all those cases the readout was done either using commercial correlator cards for up to four inputs or FPGA-based time-tagged readout of the photon streams that were subsequently processed by software correlators. There are also larger arrays available, for example [32], but with additional circuitry per pixel and a smaller fill factor.

## 2. Radhard2 SPAD array detector

We use a *Radhard2* single photon avalanche diode array with $32 \times 32$ pixels as detector for our experiments [29]. The pixels are $30 \times 30\,\mu\mathrm{m}^2$ of which only about 1.4% is active area (circular SPAD with $4\,\mu$m diameter). The photon detection probability is $\approx 30\%$ at a wavelength of 500 nm. At room temperature the dark count rate is approximately 140 Hz. After-pulsing probability is negligible at the used integration time of $\Delta t_{\mathrm{frame}} = 10\,\mu\mathrm{s}$ [33]. The pixels are row addressable and the readout of an entire frame can be done in $2.66\,\mu\mathrm{s}$. The design of the detector also allows the readout of subregions at higher speeds, e.g. a single line every $2.66\,\mu\mathrm{s}/32 = 83\,\mathrm{ns}$. Frames read from the sensor contain 1 bit of information per pixel (no photon or at least one photon in the last $\Delta t_{\mathrm{frame}}$).

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

## 3. Multi-$\tau$ hardware correlators

A hardware correlator estimates the ACF in Eq. (1) from a finite sequence of intensity measurements

$$I_n = \int_0^{\tau_{\min}} I(n \cdot \tau_{\min} + t)\,\mathrm{d}t, \quad n = 0, 1, \ldots T-1 \tag{2}$$

with the number of samples $T$ and the integration time $\tau_{\min}$ for one sample. When discretizing Eq. (1) with this intensity sequence, care has to be taken not to bias the normalization $1/\langle I \rangle_t^2$. A viable choice is the "symmetric normalization" introduced in Ref. [34]:

$$\hat{g}_{\mathrm{sym}}(\tau_k) = \frac{\overbrace{\frac{1}{T-\tau_k} \cdot \sum_{n=\tau_k}^{T-1} I_n \cdot I_{n-\tau_k}}^{=:G_{\tau_k}}}{\left[\frac{1}{T} \cdot \underbrace{\sum_{n=0}^{T-1} I_n}_{=:M_0}\right] \cdot \left[\frac{1}{T-\tau_k} \cdot \underbrace{\sum_{n=\tau_k}^{T-1} I_{n-\tau_k}}_{=:M_{\tau_k}}\right]} \tag{3}$$

with a given set of lag times $\tau_k \in \mathbb{N}$ (in units of $\tau_{\min}$, so $\tau = \tau_k \cdot \tau_{\min}$). When the full sequence $\{I_n\}_{n=0\ldots T-1}$ is available after the measurement, Eq. (3) may be evaluated directly for an arbitrary (also logarithmically spaced) set of lags $\tau_k$ in software. This gives an unbiased estimation of the ACF ("direct correlation"). To implement our hardware correlator, we use the multi-$\tau$ scheme introduced in Reference [35], which is also illustrated and compared to a linear implementation in Fig. 1. For the special case of linearly spaced lags $\tau_k = k$, a simple hardware implementation exists, which is shown in Fig. 1(a). Each of the $\Delta\tau$-blocks represents a delay of $\Delta\tau = \tau_{\min}$, which can be implemented using a flip-flop [5] clocked with a frequency of $1/\tau_{\min}$. Thus, in the final design the series of delay elements are structured as a shift register. After $n$ time-steps the input $I_{n-k}$ has propagated to the $k$-th delay element (signal $J^l_{n-k}$ in Fig. 1(a)), whereas the "undelayed signal" or "global signal" ($J^g_n$ in Fig. 1(a)) carries the current input $I_n$. So the $k$-th channel accumulates $J^l_{n-k} \cdot J^g_n = I_{n-k} \cdot I_n$ which is $G_k$ (see Eq. (3)). For the normalization additional components that accumulate the input signal $I_n$ and $I_{n-k}$ at different lags $k$ (usually called monitor channels, $M_0$ and $M_k$ in Eq. (3)) are implemented.

The multi-$\tau$ scheme uses a set of $S$ of these linear correlator blocks (Fig. 1(b,c)), with $s = 0, \ldots, S-1$. The input samples $I_{s,n}$ ($n$ is the same index as in Eq. (2)) are summed over increasingly long periods $\Delta n = m^s$, with $m = 2$ being the factor between the delay times of two subsequent blocks:

$$I_{s,n} = \sum_{k=1}^{m^s} I_{n-k}, \quad \text{for } s > 0 \tag{4}$$

with $I_{0,n} = I_n$.

Each of the linear correlators estimates the ACF at $P$ linearly spaced lags

$$\begin{aligned} \tau_{0,0} &= 0 \\ \tau_{s,0} &= \tau_{s-1,P-1} + m^{s-1} \\ \tau_{s,p} &= \tau_{s,p-1} + m^s = \sum_{i=1}^{s \cdot P + p} m^{\left\lfloor \frac{i-1}{P} \right\rfloor}, \end{aligned} \tag{5}$$

where $p = 0 \ldots P-1$.

In summary, this results in a quasi-logarithmic spacing of estimates $\hat{g}_{\mathrm{sym,multi\text{-}\tau}}(\tau_{s,p})$. The advantage of this multi-$\tau$ scheme is its simple implementation in hardware and a large dynamic

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

time range with a reasonable number of channels. Its disadvantage is a systematic error introduced by averaging: As shown in Ref. [36] the estimator $\hat{g}_{\text{sym,multi-}\tau}(\tau_{s,p})$ equals the ideal correlation function $g(\tau_{s,p} \cdot \tau_{\min})$ (see Eq. (3)) convolved with a triangular kernel with width $m^s$:

$$\hat{g}_{\text{sym,multi-}\tau}(\tau_{s,p}) = g(\tau_{s,p} \cdot \tau_{\min}) * \Lambda(\tau_{s,p}, m^s), \tag{6}$$

where $*$ denotes the convolution product and $\Lambda(\tau, \Delta\tau) = \Delta\tau - |\tau|$ for $|\tau| < \Delta\tau$ and $\Lambda(\tau, \Delta\tau) = 0$ for $|\tau| \geq \Delta\tau$, is the triangular shaped kernel.

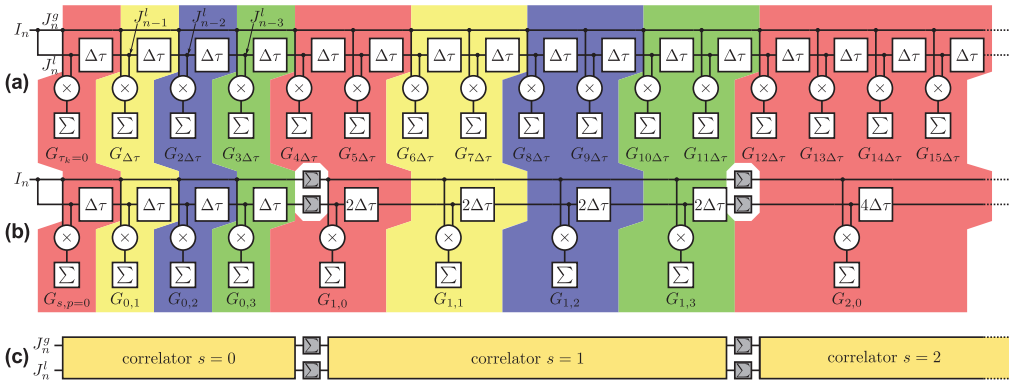


Fig. 1. Hardware design of a linear correlator (a) in comparison with a multi-$\tau$ correlator (b). Panel (c) shows a schematic view of the multi-$\tau$ correlator, where the linear correlator building blocks are summarized by a single block. Corresponding channels are grouped with the same color. Global/undelayed ($g$) and local/delayed ($l$) inputs are located on the left. For autocorrelation, the global and local signal inputs to the 0-th block ($J_n^{(g)} = J_n^{(l)} = I_n$) are identical. The $\boxed{\Delta\tau}$ blocks represent delay elements, the $\otimes$-blocks multipliers and the $\boxed{\Sigma}$-blocks accumulators.

## 4. Hardware design

Here we describe a scheme to reuse the available hardware efficiently to accommodate up to 1024 input channels. We employed a low-level hardware description language (VHDL – very high speed hardware description language) to gain speed and flexibility. This enables us to fine-tune many parameters of the final design, e.g. operational speed, memory usage, logic resource consumption and routing between logic cells.

### *4.1. Single-pixel correlator*

As shown in Fig. 1, a typical correlator is made up from channels, each corresponding to a certain lag time, and consists of a multiplier, an accumulator and a delay element.

The idea of our implementation is to use one single channel circuit to calculate all channels within one multi-$\tau$ correlator. This is possible by serial processing of the lag time channels, since the hardware in each channel is identical.

The basic arithmetic operation of one channel is to multiply and accumulate (MAC). Therefore we can map its functionality onto a MAC unit which can be found on most FPGA architectures and which is considerably faster than using generic FPGA logic cells. Only about 100 of these can be found on typical FPGAs, precluding any approach with blocks consisting of several lag channels.

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

As only one circuit is used to process all channels, we use an internal memory block (block random access memory, BRAM) to store their state (i.e., the content of the accumulator and the delayed signal). We implement this circuit by decomposing it into five steps:

1. ***L**oad* accumulator and delayed value of a channel from memory
2. ***W**ait* for memory access to complete
3. ***M**ultiply* delayed with global signal
4. ***A**dd* multiplication result to channel's accumulator
5. ***S**tore* counter and new delayed value to memory

To increase performance, these steps are executed in an interleaved manner for four channels simultaneously. This "pipelining" scheme is shown in Table 1. Our five pipeline steps are compatible with the internal pipelining of common MAC units.

Table 1. Interleaved pipeline of the linear correlator design with 8 channels.

| cycle $c$ | 0 1 2 3 4 5 6 7 | 8 9 10 11 12 13 14 15 | |
|---|---|---|---|
| ch. 0 | L W M A S | L W M A S | ch. 4 |
| ch. 1 | L W M A S | L W M A S | ch. 5 |
| ch. 2 | L W M A S | L W M A S | ch. 6 |
| ch. 3 | L W M A S | L W M A S | ch. 7 |

From one block to the next the delay time is doubled ($m = 2$) in the multi-$\tau$ scheme, making the input data rate of block $s+1$ half that of block $s$. Hence we need to execute each block only half as often as its predecessor. Thus, a complete multi-$\tau$ correlator can be executed in only twice the run-time $\Delta t_{\mathrm{lin}}$ of a single linear correlator block:

$$\overbrace{1 \cdot \Delta t_{\mathrm{lin}}}^{1^{\mathrm{st}}\text{ lin. corr.}} + \overbrace{\frac{1}{2} \cdot \Delta t_{\mathrm{lin}}}^{2^{\mathrm{nd}}\text{ lin. corr.}} + \overbrace{\frac{1}{4} \cdot \Delta t_{\mathrm{lin}}}^{3^{\mathrm{rd}}\text{ lin. corr.}} + \cdots \leq \sum_{n=0}^{\infty} \frac{1}{2^n} \cdot \Delta t_{\mathrm{lin}} = 2 \cdot \Delta t_{\mathrm{lin}} \tag{7}$$

A key requirement is that $\Delta t_{\mathrm{lin}}$ is at most half the integration time $\tau_{\mathrm{min}}$ of the input signal $I_n$.

A scheduler guarantees that a linear correlator block $s$ is only executed after its predecessor $s-1$ has been executed twice. A counter $c = 0, 1, \ldots$ is incremented with every execution of any linear correlator block. The scheduler uses the following relations to determine which linear correlator block $s$ has to be executed at a given counter value $c$ (details see appendix):

$$\begin{aligned} s = 0: &\quad c \bmod 2^1 = 0 \\ s = 1: &\quad c \bmod 2^2 = 3 \\ s \geq 2: &\quad c \bmod 2^{s+1} = (2^s - 3) \end{aligned} \tag{8}$$

Figure 2(a) shows the solution of this relation for $c$ values from 0 to 31. In the binary representation of $c$ for a linear correlator block $s$ patterns are evident that can be used to implement the scheduler efficiently. As shown in Table 2 (for $S = 8$ linear correlators), correlator block $s = 0$ is executed whenever the last bit of $c$ is $0_{\mathrm{b}}$, correlator $s = 1$ is executed when the last two bits are $11_{\mathrm{b}}$ and so forth. This scheme uses only simple comparison operations.

Between two consecutive blocks $s-1$ and $s$, adder circuitry is inserted to sum up two subsequent input signal values $I_{s-1,n-1}$ and $I_{s-1,n}$. This is done for both the delayed/local as well as the undelayed/global signal, while they are processed in the pipeline.

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

Table 2. Binary representation of counter $c$ that solves relations Eq. (8) for a given linear correlator block $s$. $\star$ denotes a don't care condition.

| lin. corr. $s$ | binary representation of counter $c$ that solves relation Eq. (8) $2^8$ | $2^7$ | $2^6$ | $2^5$ | $2^4$ | $2^3$ | $2^2$ | $2^1$ | $2^0$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | $\star$ | $\star$ | $\star$ | $\star$ | $\star$ | $\star$ | $\star$ | $\star$ | 0 |
| 1 | $\star$ | $\star$ | $\star$ | $\star$ | $\star$ | $\star$ | $\star$ | 1 | 1 |
| 2 | $\star$ | $\star$ | $\star$ | $\star$ | $\star$ | $\star$ | 0 | 0 | 1 |
| 3 | $\star$ | $\star$ | $\star$ | $\star$ | $\star$ | 0 | 1 | 0 | 1 |
| 4 | $\star$ | $\star$ | $\star$ | $\star$ | 0 | 1 | 1 | 0 | 1 |
| 5 | $\star$ | $\star$ | $\star$ | 0 | 1 | 1 | 1 | 0 | 1 |
| 6 | $\star$ | $\star$ | 0 | 1 | 1 | 1 | 1 | 0 | 1 |
| 7 | $\star$ | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |

The linear correlator as implemented here, together with its scheduler and the summation logic, is called correlation processing element (CorrPE). All channel data and intermediate summation results, the so-called pixel context, are stored in a dual-port BRAM which is associated with the CorrPE.

### *4.2. Multi-pixel correlator*

The CorrPE described above is much faster than required to calculate the ACF for a single pixel in the SPAD array. Hence, we can reuse a single CorrPE for multiple pixels by switching between pixel contexts. This "pixel scheduler" uses a double-buffering strategy. While a CorrPE operates on the current context, the previous context is exchanged with the next context to be processed. The pixel contexts are stored in external background memory (SRAM), because they exceed the capacity of the internal memory. In Fig. 3 we illustrate how we reuse one CorrPE for several pixels in comparison to a naïve implementation using one CorrPE per pixel.

In addition to the channel data, the cycle counter $c$ and an accumulator for the local and the global input signals have to be saved. The latter are used for normalization and cross-correlation.

A single CorrPE is used to process an entire column (ACFs of $n_y = 32$ pixels) of our SPAD array. To handle the full $n_x \times n_y$ array of pixels, we instantiate $n_x = 32$ CorrPEs in parallel. An overview of this scheme is shown in Fig. 4. A "data acquisition" circuit communicates with the

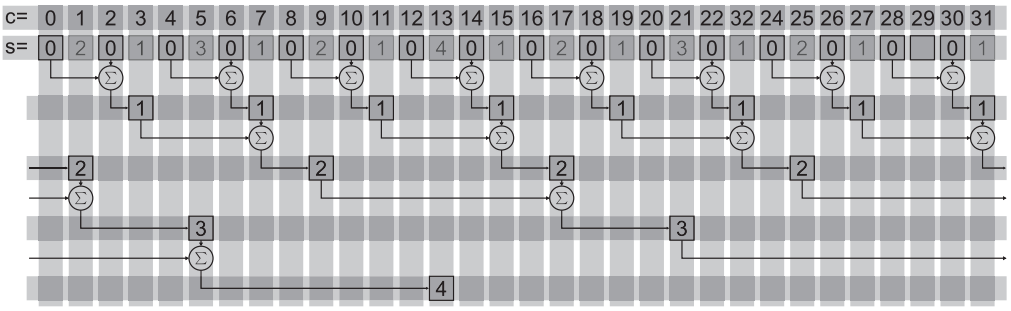


Fig. 2. Block scheduling algorithm. The counter $c$ in the first row defines the current cycle. The second row shows the block $s$ that is processed in the corresponding cycle. Further below the order of processing is shown: When a block $s$ has been processed twice, the following block, $s+1$, can process the sum of the two preceding $s$-blocks.

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

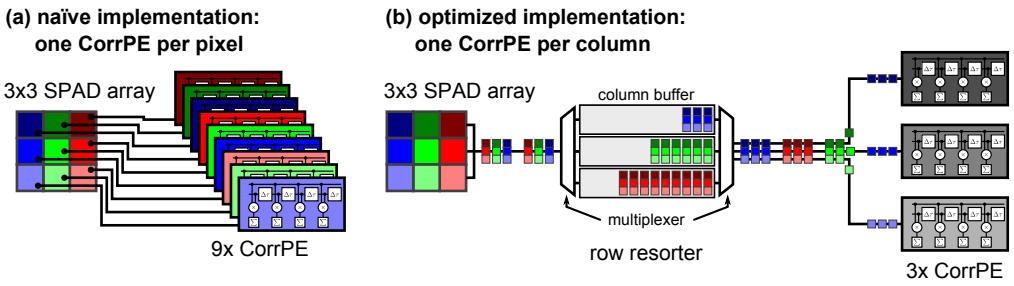


Fig. 3. Comparison of a naïve implementation (a: one correlator per pixel) and an optimized implementation (b: reuse CorrPEs for several pixels) of a multi-pixel multi-$\tau$ correlator for a $3 \times 3$ SPAD array.

SPAD array and provides the image data for the correlators. As the image data is streamed out row by row, and each of the $n_x$ CorrPEs is only processing data from one specific context (i.e. a specific row), the remaining pixels have to be buffered row wise in 32 FIFOs (first-in first-out memory buffer) localized in external RAM.

In addition, our design contains two USB 2.0 interfaces. One is used to send the raw data stream from the SPAD array to the computer, which allows further data processing. We also use these raw data to verify our correlator design. Via the second interface, intermediate and final results from the correlators are transferred to the host computer. The intermediate results allow implementing a live view of the ongoing calculation of the ACFs.

### *4.3. ACF normalization*

The intensity values in the denominator of Eq. (3) are typically obtained from *monitor channels* $M_{\tau_k}$, which accumulate the total photon count at a given lag time $\tau_k$. In contrast to other implementations, we use only one monitor $M_0$ (input signal accumulator) per multi-$\tau$ correlator. Symmetric normalization (see also Eq. (3)) yields the following:

$$\hat{g}_{\text{sym,multi-}\tau}\left(\tau_{s,p}\right) = \frac{G_{\tau_{s,p}}}{2^s} \cdot \frac{T}{M_0 \cdot M_{\tau_{s,p}}} \tag{9}$$

After measuring $T$ samples, the number of samples that have propagated through the corre-

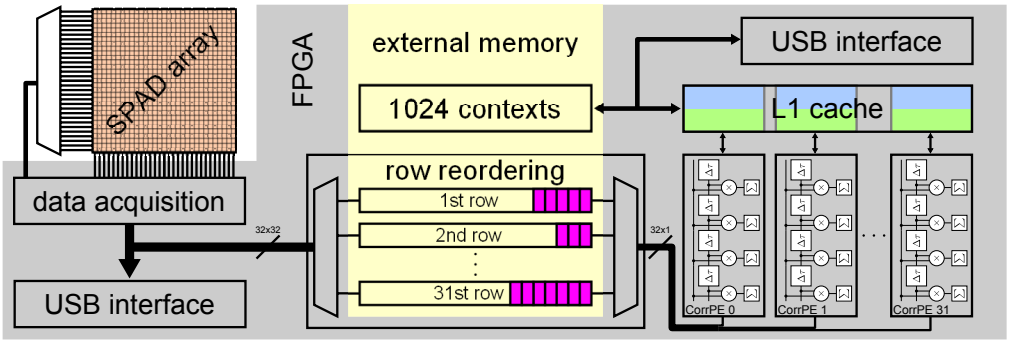


Fig. 4. System layout and data path - from data acquisition to correlation. One USB interface is used to stream the raw images, the other for streaming (intermediate) results. The first level cache (L1, double buffered) is used to hold the context of the currently processed pixel. FIFOs for row resorting and for context storage use external memory.

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

lator to a distinct channel $(s,p)$ is $T-\tau_{s,p}$. Under the assumption that $I_n$ is a stationary random process and that $T>\tau_{s,p}$, the per-channel monitors $M_{\tau_{s,p}}$ can be estimated in the following manner:

$$M_{\tau_{s,p}} = M_0 \cdot \frac{T-\tau_{s,p}}{T}. \tag{10}$$

This normalization and model fitting is done on the host computer, since floating-point arithmetic cannot be implemented efficiently on an FPGA. Although the correlation on the FPGA and the normalization on the host computer can easily be done in real time, the fits in most cases cannot. Usually a single curve fit takes tens of milliseconds, thus fitting all 1024 ACFs would amount to $\geq 10\,\mathrm{s}$ additional computing time; but this still allows to display fit parameter maps (images) within an acceptably short delay. In current imaging FCS software systems [37] the data is loaded and correlated on the host computer, which for a measurement of typically 10 s takes a couple of minutes for 1024 pixels at the frame rate of our sensor.

To show that the estimation in Eq. (10) yields good results, we simulated different correlator types in software. The results for a direct estimation of the ACF using Eq. (3) (green), a multi-$\tau$ correlator with a monitor channel per lag (blue) and our estimation (magenta) can be seen in Fig. 5, where the data in (a) and (b) were obtained by correlating the input signal $I(t) = 1+\sin(2\pi t/(1.51\cdot 10^{-4}))$ for which the exact ACF is known to be $g^{(\mathrm{theoretical})}(\tau) = 1+\cos(2\pi\tau/(1.51\cdot 10^{-4}))$ (time $t$ and lags $\tau$ are unit free). The data in Fig. 5(c) was created by simulating a $T_{\mathrm{sim}} = 1\,\mathrm{s}$ long FCS experiment with one diffusing species [38]. It was computed with our FCS simulation software described in Ref. [39]. Further details on the simulation code are shown in the appendix.

For short lags the estimated ACFs resemble the theoretical curves quite well. Multi-$\tau$ correlators have an increased absolute error for longer lags, which is due to the averaging described in Eq. (6). This can be seen especially in the case of the sine wave signal. The multi-$\tau$ estimates can still be used for FCS experiments, as here the ACFs usually decay to 1 (white noise) for large lag times, and thus the systematic error drops to zero again (for a detailed discussion of this, see e.g. Ref. [36]). For $\tau \gtrsim T_{\mathrm{sim}}/10$, multi-$\tau$ correlators show additional systematic deviations from the theoretical curve and from the direct estimation, because the channels are not averaged over sufficiently many samples to yield reliable results. Here the multi-$\tau$ implementation with multiple monitor channels performs better due to the better estimation of the normalization factor $M_{\tau_{s,p}}$.

### *4.4. Crosscorrelation (CCF)*

Our design can also calculate cross-correlation:

$$g^{(x,y)}(\tau) = \frac{\left\langle I^{(x)}(t)\cdot I^{(y)}(t+\tau)\right\rangle_t}{\left\langle I^{(x)}(t)\right\rangle_t \cdot \left\langle I^{(y)}(t)\right\rangle_t}$$

between two input signals $I^{(x)}(t)$ and $I^{(y)}(t)$ at the local $J^{(\mathrm{l})}$ and global $J^{(\mathrm{g})}$ inputs of the CorrPE (see Fig. 1 where both signals are tied to $I(t)$ for ACF calculation). This changes the multi-$\tau$ estimator Eq. (9) to:

$$\hat{g}^{(\mathrm{CCF})}_{\mathrm{sym,multi\text{-}\tau}}(\tau_{s,p}) = \frac{G_{\tau_{s,p}}}{2^s}\cdot\frac{T}{M_0^{(g)}\cdot M^{(l)}_{\tau_{s,p}}} \tag{11}$$

Here the normalization uses the monitors $M_0^{(g)}$ for the global and $M_0^{(l)}$ for the local signal ($M^{(l)}_{\tau_{s,p}} = M_0^{(l)}\cdot(T-\tau_{s,p})/T$). For autocorrelation (see above) only one monitor channel is needed, as $M_0 = M^{(g)} = M^{(l)}$.

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

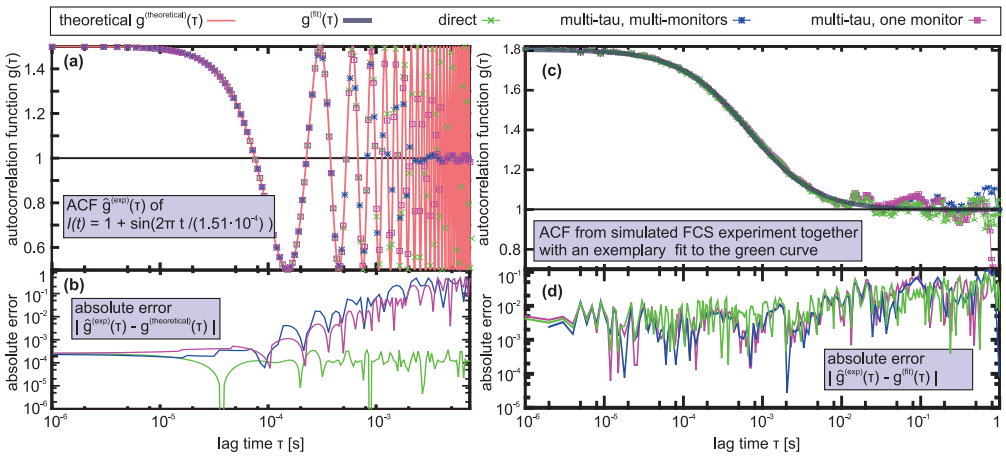


Fig. 5. Simulation results for different implementations of multi-$\tau$ correlators: The left panels (a,b) show simulations for a sine wave input signal $I(t) = 1 + \sin(2\pi t/(0.151\,\mathrm{ms}))$. The simulations on the right (c,d) were created using an FCS simulation. The top graphs (a,c) are estimates of the autocorrelation function using direct correlation from Eq. (3) (green), a multi-$\tau$ correlator with one monitor channel per lag time (blue) and our estimated normalization from Eq. (10) (magenta). Graph (a) also shows the theorectical ACF $g^{(\mathrm{theoretical})}(\tau) = 1 + \cos(2\pi\tau/(0.151\,\mathrm{ms}))$ for the sine signal (light red). The lower graphs (b,d) show the absolute deviation of the estimates from $g^{(\mathrm{theoretical})}(\tau)$ (b) and from a fit to the curves (d). The parameters resulting from the fit in (c) are the same within $< 2.5\%$ for all three estimates.

## 5. Performance & implementation details

The complete design is implemented in two Virtex-II Pro FPGAs (XC2VP40, Xilinx, http://www.xilinx.com/, San Jose, USA), on a LASP development board [40]. The first FPGA is used for data acquisition and line reordering, while the other one is used to implement the correlators. The total resource consumption within the second FPGA is around 40%. Using this hardware platform, there are currently $P = 8$ channels within each of the $S = 14$ linear correlator blocks.

In Eq. (7) we showed that the complete multi-$\tau$ correlator can be executed within twice the time needed for the first linear correlator block, so we devote half of the execution time to the first linear correlator and the rest to the remaining blocks. Therefore a new input sample can be accepted only once every $2\Delta t_{\mathrm{lin}}$. Here $\Delta t_{\mathrm{lin}} = 2P + 3$ cycles is the time needed to process a new input sample $I_n$ in the first linear correlator block. The 3 additional cycles are used for data hand over to the next block.

The CorrPEs run with a clock frequency of 144 MHz, which corresponds to an execution time of 264 ns $= 2\Delta t_{\mathrm{lin}}$ for a single input signal. This is the minimum timespan between two subsequent samples, if no pixel multiplexing is used. Hence, our current FPGA platform can calculate 32 different ACFs or CCFs with a minimum lag time of 264 ns, which is comparable to the 100 ns designs presented in Ref. [14] and more recently in Ref. [19]. Trading time resolution for more correlation functions, we can process all 1024 pixels of the SPAD array at a frame rate of 100 kfps in real time or regions of interest with $< 32$ lines with even higher frame rates.

To estimate the memory consumption of our design, we first look at the pixel context, which consists of 128 words of 64 bits each. Table 3 shows a detailed memory layout. Sixteen of the upper 32 bits of the raw accumulators store the current value of the delay registers $J^{(l)}_{s,p}$.

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

The lower 32 bits contain the accumulator $G_{\tau_{s,p}}$. The monitor channels are 32 bits each. Data handover between consecutive blocks (accumulated local and global signals) is done via 16 bit-wide memory areas, which is sufficient for $S \leq 16$. Since in the later linear correlators the accumulated input signals $I_{s,n}$ are multiplied, the sums $G_{\tau_{s,p}}$ and also the $I_{s,n}$ can get relatively large and may not fit in the 32 bit memory locations available. However, due to constant streaming of intermediate results to the host computer, counter overflows can be detected and corrected.

Table 3. Memory layout of a pixel context

| **data word** | **usage** |
|---|---|
| 0...111 | delay register $J_{s,p}^{(l)}$ and raw accumulator values $G_{\tau_{s,p}}$ |
| 112 | status counter $c$ |
| 113...125 | intermediate results for accumulated input signals $I_{s,n}$ |
| 126 | global monitor channel $M_0^{(g)}$ |
| 127 | local monitor channel $M_0^{(l)}$ |

For our $32 \times 32$ pixel SPAD array the pixel contexts are stored in $32 \cdot 32 \cdot 128 \cdot 64$ bits $=$ 512 KBytes of external SRAM. A second SRAM stores the 256 KBytes used for the FIFOs (2048 entries each) that hold the pixel data until they are processed.

The correlator design is implemented in VHDL (very high speed integrated circuit hardware description language). It can be configured via generics (a VHDL feature). Thus we can reuse our design for different needs (e.g. for different sensor sizes and frame rates). We expect our design to show a significant increase in performance when implemented on newer FPGA generations (e.g. Virtex 5 or Virtex 6 from Xilinx).

## 6. Benchmark experiments

To ensure the functional correctness of the designed correlator, a simulation of the hardware was tested using random input data. The outcome was compared to the results of a software implementation of the multi-$\tau$ correlator using the same data set. Both yielded exactly the same results. The comparison was also done using real experimental data from the SPAD array. Again, both results were identical.

To demonstrate the functionality of the whole system, we tested the design using the SPAD array to record an LED connected to a sine wave generator set to 2.5 kHz. Figure 6 shows the results of this measurement. A fit to the data recovered the chosen frequency.

Finally we tested our system in a custom selective plane illumination microscope (SPIM), comparable to the setup described in [41]. A 491 nm beam from a DPSS laser (Calypso, Cobolt AB, Solna, Sweden) is formed into a light sheet of 1.25 $\mu$m width ($1/e^2$ half-width) by a Nikon Plan Fluor 10x/NA0.3 microscope objective and cylindrical lens of focal length $f = 100$ mm (CKX18-C, Newport Spectra-Physics GmbH, Darmstadt). A Nikon CFI Apo-W NIR 60x/NA1.0 water dipping objective and an achromatic lens with $f = 100$ mm (AC254-100-A-ML, Thorlabs GmbH, Dachau, Germany) are used to image the acquired fluorescence onto the SPAD array *Radhard2*, with a 30$\times$ magnification. A 500nm long-pass filter (Edge Basic 488LP, Semrock, Rochester, USA) suppresses scattered light by about a factor of $10^{-6}$ at 491 nm. Figure 7 shows the results of an experiment with fluorescent microspheres of diameter 40 nm (Invitrogen FluoSpheres YG carboxyl-modified, Life Technologies GmbH, Darmstadt). According to the manufacturer, each bead has a brightness which is equivalent to 350 fluorescein molecules. They were dissolved in water (1 : 1000 dilution from stock concentration

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

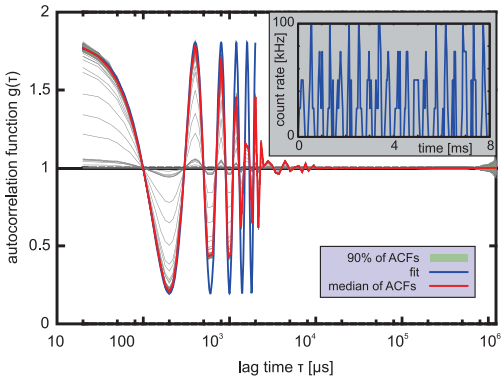


Fig. 6. Distribution of 992 (gray) ACFs taken by our sensor (first 31 columns only), exposed to a 630 nm LED sine-modulated with a frequency of 2.5 kHz. The inset shows a section of the input count rate summed over 4 samples or 40 $\mu$s for each point. The correlator was running for 1.2 s (131072 samples at $\tau_{\mathrm{min}} = 10\,\mu$s). The gray curves with significantly lower amplitude are due to hot pixels of the SPAD array; since these SPADs fire randomly, the corresponding correlation amplitude is decreased. The median, which tends to be less sensitive towards outliers than the mean, is shown in red. The theoretical model $g^{(\mathrm{theoretical})}(\tau) = 1 + A \cdot \cos(2\pi f \cdot \tau)$ is fitted against the median in the interval $[10\,\mu\mathrm{s}, 1\,\mathrm{ms}]$, and is shown until $\tau = 2\,$ms. The fit yields a frequency of $(2502 \pm 4)\,$Hz. Compare also Fig. 5(a).

$c_{\mathrm{stock}} \approx 2.4\,\mu$M) and were mounted in a small sample bag made from 25 $\mu$m thin transparent foil matching the refractive index of water (LUMOX FOLIE 25 M, SARSTEDT AG & Co, Nümbrecht). The sample was illuminated with a laser power of about 2.33 mW, as measured behind the projection objective. This power is distributed over the light sheet of height 4 mm and center $1/e^2$-width of 2.5 $\mu$m, which amounts to an intensity in the focus of about 23.3 W/cm$^2$.

For each ACF a fit to the standard FCS model function for 3-dimensional free diffusion was perfromed:

$$g^{(\mathrm{fit})}(\tau) = \frac{1}{N} \cdot \left(1 + \frac{\tau}{\tau_D}\right)^{-1} \cdot \left(1 + \frac{\tau}{\gamma^2 \tau_D}\right)^{-1/2} \tag{12}$$

Here $N$ is the average particle number inside the focus and $\tau_D$ is the diffusion decay time. The parameter $\gamma$ describes the aspect ratio of the Gaussian focus. The (unweighted) fits were performed using a Levenberg-Marquardt least squares fitting routine (lmfit [42], implemented in our FCS data evaluation package QuickFit 3.0. [43]). The model Eq. (12) is justified for a SPAD array and the described SPIM setup, as each SPAD effectively operates as a single pinhole. The longitudinal and lateral size of the point-spread function were measured by scanning beads, embedded in a clear and low-scattering, but stiff gel (0.5 % Phytagel, Sigma-Aldrich, Seelze, Germany), through the image plane in respective directions. From this measurement we obtained a lateral $1/e^2$ half-width of $w_{xy} = (0.5 \pm 0.2)\,\mu$m and longitudinal half-width of $w_z = (0.8 \pm 0.1)\,\mu$m and therefore an aspect ratio of about $\gamma = (1.60 \pm 0.72)$, which was fixed in the fits. The lateral and longitudinal widths fit the theoretically expected values for the setup, if the depth-selection due to the pinhole is taken into account. Also we assume a Gaussian detection probability distribution, as usually done in SPIM-FCS [27,28].

The average diffusion time was estimated from the median of the distribution of all diffusion times (see Fig. 7(e)) to be $\tau_{\mathrm{D}} = 10.2\,$ms. From this we can calculate a diffusion coefficient of $D = w_{xy}^2/(4\tau_{\mathrm{D}}) = (6.1 \pm 2.5)\,\mu\mathrm{m}^2/\mathrm{s}$, which compares well to the value of $(5.9 \pm 1.5)\,\mu\mathrm{m}^2/\mathrm{s}$ measured with confocal FCS for the same sample. Due to the Gaussian nature of the light

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

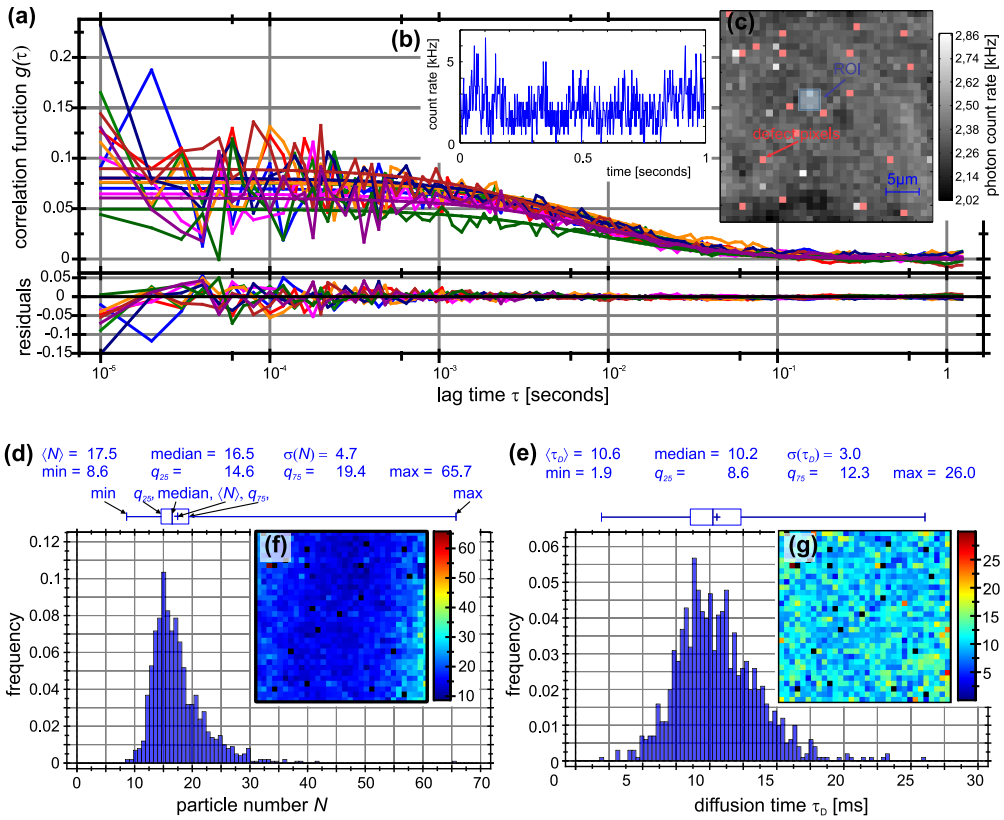


Fig. 7. Results from SPIM-FCS measurements of fluorescent microspheres with diameter 40 nm. The measurement duration was 20.97 s.
*(a)* A set of nine example ACFs, the according fits and residuals from the marked region of interest. *(b)* example timetrace of one pixel, averaged over 2 ms or 200 samples for each plotted point. *(c)* summed intensity image. *(d)* histogram of the particle number $N$ from the fits. In addition some statistical characteristics of the histogram are given (average $\langle N \rangle$, median, standard deviation $\sigma(N)$, minimum/maximum value and 25%/75% quantile $q_{25}, q_{75}$). *e* histogram of the diffusion time $\tau_D$ from the fits. *(f)* particle number map. *(g)* diffusion time map in units of milliseconds.

sheet, its width increases with the distance to the focal line. This leads to an increased number of detected particles $N$ at the edges of the field of view (see Fig. 7(f)). The average count rate during the measurement was $\langle I \rangle = 2400\,\text{Hz}$ (see also Fig. 7(b) and (c)), which is well above the dark count rate (DCR) of the *Radhard2* sensor of about 140 Hz [29]. The overall background countrate (including the DCR) was $I_{\text{background}} \approx 300\,\text{Hz}$ during the measurement, which leads to a correction of the measured particle number (as shown in Fig. 7) of $N_{\text{real}} = N_{\text{measured}}/(1 + I_{\text{background}}/\langle I \rangle)^2 = 0.79 \cdot N_{\text{measured}}$ [44]. Thus from the corrected median particle number of median$(N_{\text{real}}) = 13.0$ we get a molecular brightness of about 184 Hz/particle, which is about a factor of 530 less than in the measurement on our confocal setup ($NA = 1.2$), where the average excitation intensity in the focus was about a factor of 250 higher than in our SPIM setup. The difference can be explained by the higher NA, quantum efficiency and larger pinhole size of the confocal setup.

In Fig. 8 we compare measurements for particles of different sizes (Invitrogen FluoSpheres

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

YG carboxyl-modified, 40 nm and 100 nm diameter, also diluted in water). These examples were selected to show the functionality of our correlator and readout system. A more detailed characterization of SPAD arrays for SPIM-FCS applications will be the topic of a follow up publication. Our current SPAD array is limited in its applicability to high intensity samples, due to its small fill factor. We will address this problem with next generation sensors that are equipped with microlenses.

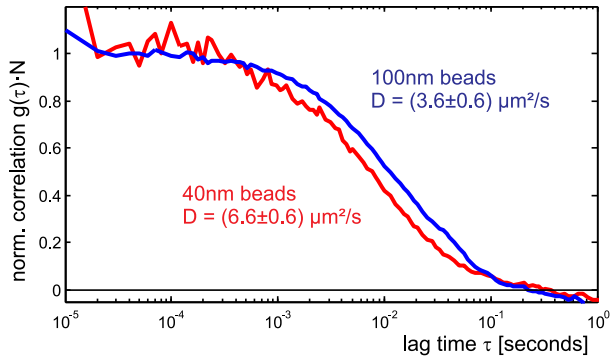


Fig. 8. Example normalized correlation curves for two different sizes of beads, with diameters 40 nm (red) and 100 nm (blue). The curves are an average over 16 single-pixel ACFs each and the given diffusion coefficients are the average and standard deviation from fits to these 16 single curves.

## 7. Conclusion

In this paper we presented the implementation of an FPGA-based multi-$\tau$ correlator design that can calculate 1024 correlation functions in real time at a minimum lag time of 10 $\mu$s. To our knowledge this is the largest number of real time multi-$\tau$ correlators implemented so far in a single device. The minimum lag time of 10 $\mu$s in our design is longer than that of currently available hardware correlators (e.g. from ALV GmbH, Langen, Germany or correlator.com, Bridgewater, USA and Reference [11]), but those are limited to at most 32 auto-correlators.

Recently, Mocsár et. al. proposed a hardware correlator implementation with four input channels, comparable to commercially available multi-channel correlators (e.g. from ALV GmbH, Langen, Germany) using an FPGA card from National Instruments, with a minimal sampling time of 400 ns [19]. That approach reuses a single linear correlator core for different blocks within the multi-$\tau$ correlation scheme and for the multiplexed processing of all four input signals. Due to the utilization of a high level description of the hardware using the well established LabView software (National Instruments), their system is easy to adapt to different experimental requirements even for non-experts. But this approach also limits the overall performance in terms of resource consumption and speed of execution. Jakob et. al. proposed to use a single multiply-accumulate cell (MAC), to calculate all correlation channels in serial manner [14], the so called "virtual correlator architecture". Our design further extends and optimizes both of these hardware reuse schemes: Only a single MAC per correlator is used, which allows us to implement 32 parallel correlators within a single Virtex-2 FPGA. These can then process up to 1024 pixels or input signals, using an advanced multiplexing scheme.

We use our design to correlate the output of a single-photon avalanche diode (SPAD) array used as image sensor in a selective plane illumination microscope (SPIM). This combination allows us to perform imaging fluorescence correlation spectroscopy at the 10 $\mu$s time scale, which covers the range of motion of small molecules in solution and living cells. We presented

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012

several benchmark experiments that show the applicability and functionality of our correlator system. The low photon detection probability due to the small fill factor of our SPAD array will be addressed with next-generation sensors equipped with microlenses.

Our correlator design is flexible and extensible: besides estimating temporal autocorrelation functions, also cross-correlation functions between different pixels or multiple colors (using spectrally resolved detectors) can be calculated. To achieve a larger dynamic range (e.g. for long measurements of $\tilde{T} \gg 1\,\mathrm{s}$) it is possible to output the accumulated global and delayed values after the last block. As the data rate is reduced by a factor of $2^{14}$, then the correlation can easily be processed on the host computer.

The entire hardware correlator is limited to process input values at a rate of $1024 \cdot 100\,\mathrm{kHz} = 102.4\,\mathrm{MHz}$. Hence, if the whole array is read, we can achieve a minimal lag time of $10\,\mu\mathrm{s}$. For a region of interest containing 64 pixels (2 lines), a frame rate of $102.4\,\mathrm{MHz}/64 = 1.6\,\mathrm{MHz}$ and therefore a minimal lag time of $\tau_{\mathrm{min}} = 0.625\,\mu\mathrm{s}$ can be achieved. As our current implementation accepts up to 16 bit wide inputs, we can overcome the clipping due to the one bit counters within our sensor: The data acquisition logic will be extended to accumulate consecutive frames. A valid choice are 2 bit counters, that allow the sum of three subsequent images. This results in a possible full frame integration time of $3.33\,\mu\mathrm{s}$, which is slightly above the sensor's limit of ($2.66\,\mu\mathrm{s}$). Newer FPGAs in combination with next-generation sensors, will allow us to increase the processing speed even further.

**Acknowledgments**

The project was supported by a NUS-BW (National University of Singapore / Baden-Württemberg) joint grant to J.L., a doctoral fellowship of the Helmholtz International Graduate School for Cancer Research to J.B., a doctoral fellowship of the Heidelberg Graduate School of Mathematical and Computational Methods for the Sciences to J.W.K. We thank Xilinx, San Jose, USA for donating the FPGAs on the LASP development board. G.V. receives support by the German-Hungarian program for the exchange of researchers by the German Academic Exchange Service and the Hungarian Scholarship Board (MÖB-47-1/2010) OTKA K77600 and TAMOP 4.2.1/B-09/1/KONV-2010-007.

#165180 - $15.00 USD Received 22 Mar 2012; revised 1 Jun 2012; accepted 21 Jun 2012; published 20 Jul 2012