Flex99-12C USB correlator interface

Marcel Leutenegger

École Polytechnique Fédérale de Lausanne
Laboratoire d'Optique Biomédicale
1015 Lausanne, Switzerland

October 25, 2005

User manual for the graphical user interface to the Flex99-12C USB correlator. The user interface runs either as an executable binary or as a MATLAB script under Microsoft® Windows.

Contents

1 Introduction

The Flex99OEM-12C USB hardware correlator, sold by correlator.com, was just equiped with a sample C/C++ recording program. This software has been written to gain user-friendly access to the correlator within MATLAB scripts.

Features

- Interactive control of the correlator function and automatic shutter control.
- Live display of the correlation curve and the intensity trace(s).
- Afterpulsing estimation and suppression. Even with an APD-based single photon counting module, the useful delays for parameter extraction extend to less than one microsecond.
- Automated data export to a MAT-file or into the Carl Zeiss®/Evotec® Confocor file format.
- Integrated parameter extraction for alignment feedback and live control.
- Customizable parameter extraction, data export and shutter control.

System requirements

- Correlator.com® Flex99OEM-12C USB hardware correlator and USB driver.
- A Pentium™ III class or better processor.
- 16MByte of free RAM and a free USB1.1 port.
- Microsoft® Windows 95/98/2000/XP operating system.
- Mathworks MATLAB® R12 or newer to run the script, or the Mathworks MATLAB® graphics library to run the binary.

2 Installation

2.1 Script

1. Install the Flex99OEM-12C USB driver.
2. Download the Flex99-12C correlator scripts from MATLAB-Central or the author's homepage. Unpack the archive into a folder of your choice. Copy the SDK library **Flex99-12C.dll** or **Flex99OEM-12C.dll** into the **private** folder and rename it to **Flex99-12C.dll**.
3. Plug in the correlator and execute **flex99.m** from the MATLAB command window. If the software is installed in a folder that is not part of the MATLAB path, you have to change to that folder first.

2.2 Binary

1. Install the Flex99OEM-12C USB driver.
2. Download the Flex99-12C correlator binaries from the author's homepage. Unpack the archive into a folder of your choice. Copy the SDK library **Flex99-12C.dll** or **Flex99OEM-12C.dll** into this folder and rename it to **Flex99-12C.dll**. Install the the MATLAB 6.0 graphics library.
3. Plug in the correlator and start **Flex99-12C.exe**.

3 User interface

Figure 1 shows the main user interface.

The upper-left graph displays the intensity of channel A (blue) and, in case of a cross-correlation, channel B (red). During a correlation, the axis label shows the elapsed time and the current count rate(s). Use the context menu for hiding one of the intensity traces or clearing them. The lower-left graph displays the correlation and the fit curve(s). You can zoom into the correlation graph with a mouse drag. A mouse click brings you back to the default display. The graphs are refreshed at most every 100ms.

With the panel at the right, you control the function of the correlator. The following subsections outline the user controls.

3.1 Correlation type

Configures the correlator either for an auto-correlation, channel A × A, or for a cross-correlation, channel A × B. You can switch the correlation type at any time.

3.2 Integration time

Defines the interval before the correlator starts a new correlation. You can increase/decrease the intervals with the spinners or directly enter a value. The effective integration time is typically 50ms longer than the value entered.

The **live operation** integration time defines the interval used in*live mode* during alignment. **On recordings** sets it when saving the correlation to a file in *snap mode*.

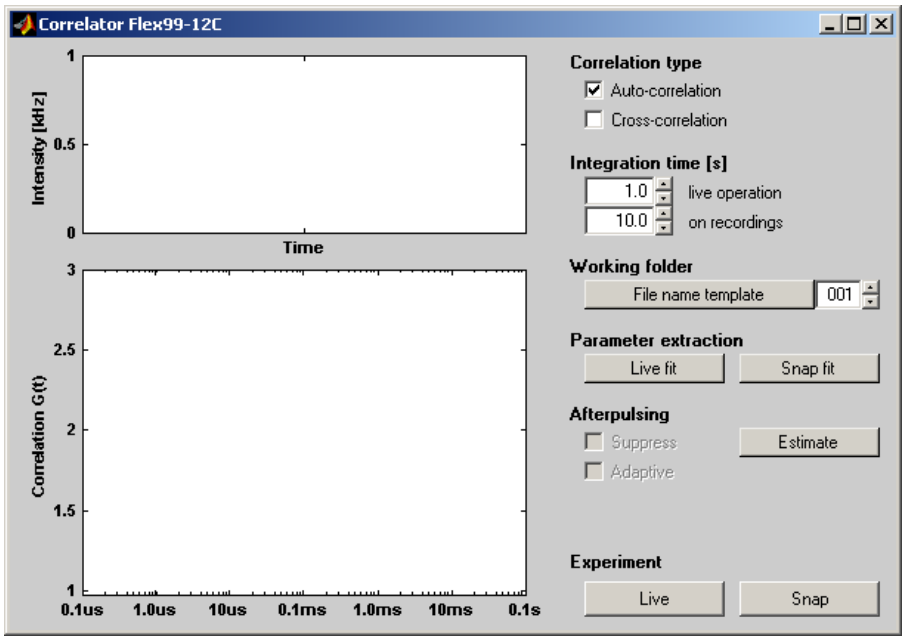

Figure 1: Main window of the graphical user interface.

3.3 Working folder

Sets the file name template and the running number used for automated data export. Clicking on the toggle button displays a file dialog to define the export folder and the file name template. Accepting the file name template activates the automatic file name generation. Clicking again on the toggle button turns it off.

The file names generated are

$$(3 \text{ digits} \in [001, 999])\ \textit{template.extension}$$

3.4 Parameter extraction

Only available with the script version.

Two toggle buttons activate fitting of the correlation curve to extract the parameters. **Live fit** is used in *live mode*. The fitted correlation curve (blue) overlays the measured correlation (black). **Snap fit** is used in *snap mode* and the fitted correlation is displayed in red. The extracted parameters are written to the command window.

3.5 Afterpulsing

Avalanche Photo Diodes APDs can have a significant afterpulsing[1] probability. Afterpulsing shows up as an exponential increase of the correlation curve for small delay times. The amplitude decreases with increasing count rate.[2] Cross-correlating the signal of two Single Photon Counting Modules SPCMs avoids afterpulsing at the expense of buying a second detector.

This interface integrates a numerical reduction of the afterpulsing contribution to the auto-correlation. You can estimate the afterpulsing contribution by recording the correlation of an uncorrelated signal of constant brightness (preferably day light). The integration time for an estimation should be at least 100s or 10 times longer than for the experiments. The mean count rate $\langle I \rangle$ should be close the the count rate during your experiment.

It is recommended to estimate the afterpulsing contribution for various intensities. For example, you may estimate it for

$$\langle I \rangle \in \{5\text{kHz},10\text{kHz},20\text{kHz},40\text{kHz},...,2\text{MHz}\}$$

Afterpulsing estimations are automatically stored in the **afterpulse.mat** file in the current folder. Please keep in mind that the correction is only valid for the filter set and the SPCM used during the estimation.

Estimation

1. Set-up the integration time.
2. Remove any fluorescent sample and any source of fluctuation.
3. Run the correlator in live mode and adjust the intensity level.
4. Start the estimation by clicking on **estimate**.

[1] An afterpulse is the delayed release of a trapped electron leading to two correlated pulses due to one detected photon.
[2] A high count rate increases the chance of masking an afterpulse event by another photon event.

5. On finish, check if the correlation curve is smooth and goes to unity without oscillations for delay times above 10μs. If anything is fine, accept the estimation. Be carefull not to corrupt the **afterpulse.mat**.[3]

Correction *Only available if at least one estimation was recorded.*

Enable the afterpulsing correction with the **suppress** checkbox. The auto-correlation should tend to a constant for delay times below 1μs. Typically, the correction increases noise at these values.

Adaptive correction *Only available if at least one estimation was recorded.*

Enabling the **adaptive** checkbox activates automatic intensity matching of the afterpulsing estimation. The interface reloads all afterpulsing estimations from *afterpulsing.mat*, calculates the mean contribution and adapts the estimation automatically[1] to the mean count rate of the current measurement.

3.6 Experiment

Start and stop the hardware correlator.

A click on **live** runs a correlation during the *live mode* integration time, fits the correlation if *live fit* is active and repeats these steps continously. To stop the correlator, click again on the button, now labeled **stop**.

With **snap**, you start recording a correlation for at most the *snap mode* integration time. On recording, the control panel is disabled except the **snap** button itself, now labeled **save**. Clicking on **save** immediately terminates the current take. If *snap fit* is active, the parameters are extracted. If a *file name template* is active, the data is automatically exported and the *running number* is incremented. Otherwise, you will be asked for a file name.

4 Errors

Error messages are displayed in an alert box shown in figure 2. The alert box does not block the user interface but gets highlighted on each error. Therefore, you may move it out of the main interface without closing it.

Typical error messages and potential causes are listed thereafter.

Could not load the Flex99-12C USB interface.

Indicates that the MATLAB interface did not find the USB library **Flex99-12C.dll**. The library has to reside in the program folder if you use the binary, respectively in the **private** folder if you use the scripts. You may have to rename the **Flex99OEM-12C.dll**.

[3] In case of a mistake, you can remove it manually anyway.

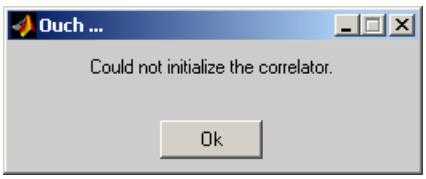

Figure 2: Alert box displaying a typical error message. The alert box can be moved out of the main interface without closing it.

Could not load all USB interface functions.

Strange – do you use the correct USB library **Flex99OEM-12C.dll** or **Flex99-12C.dll**? Indicates that the **Flex99-12C.dll** was loaded but did not contain all functions called by the interface.

Could not initialize the correlator.

Is the USB correlator connected to your computer? After connection, the driver needs a couple of seconds to recognize it. If the problem persists for more than one minute, try to remove the correlator and connect it again. Some computers have difficulties to recognize the correlator if hot-plugged.

5 Extensions

The following features are only available in the script version.

The data export, parameter extraction and shutter control can be easily adapted to your needs. Just modify the scripts **export.m**, **fitcorr.m** and **shutter.m** in the **private** folder of the correlator interface.

5.1 Data export

Whenever you export the correlation data, **export.m** is called to write the data into a file.

Arguments

file	`char`	Name of file to write
correlation	`double`$[1 \times N]$	Auto- or cross-correlation
delay	`double`$[1 \times N]$	Correlation delays [s]
intensity	`double`$[2 \times M]$	Intensity traces [Hz]
time	`double`$[1 \times M]$	Trace sample times [s]
mode	`char`	Correlation mode {A × A,A × B}

Original file export.m

```
%Save the correlation and intensity traces to a MAT-file.
%
function export(file,correlation,delay,intensity,time,mode)
if numel(file) < 4 | ~isequal(file(end-3:end),'.mat')
   file=[file '.mat'];
end
author='Leutenegger Marcel  25.10.2005';
origin='MATLAB interface to Flex99-12C correlator';
date=uint16(datevec(now));
step=pow2(12.5e-9,23);
save(file,'-mat');

%Save the correlogram in the Zeiss/Evotec format.
%
% f      File name
% g      Correlation
% t      Delays [s]
% I      Intensity [Hz]
% T      Sample times [s]
%
% function export(f,g,t,I,T,m)
% if numel(f) < 4 | ~isequal(f(end-3:end),'.dat')
%    f=[f '.dat'];
```

```
% end
% f=fopen(f,'w');
% fprintf(f,'##TITLE=%s FCS data\n##JCAMP-DX=4.24\n##ORIGIN=Flex99-12C correlator - MATLAB interface by
     Leutenegger Marcel  25.10.2005\n',m);
% fprintf(f,'##DATE=%s\n##TIME=%s\n##DATA TYPE=FCS Correlogram\n##NPOINTS=%d\n##XYPOINTS=(XY..XY)\n',
     datestr(now,1),datestr(now,15),length(g));
% fprintf(f,'%g\t%g\n',[1e3*t g].');
% [m,n]=size(I);
% fprintf(f,'##DATA TYPE=FCS Count Rates\n##NPOINTS=%d\n##DELTAX=0.104831\n##XYPOINTS=(XY..XY)\n',m);
% s='%g\n';
% while n > 0
%    s=['%g\t' s];
%    n=n-1;
% end
% fprintf(f,s,[pow2(12.5e-9,23)*(0:m-1);1e-3*I.']);
% fprintf(f,'##END=FCS data\n');
% fclose(f);
```

5.2 Parameter extraction

The function **fitcorr.m** implements a fit model and extracts the parameters. It displays the fitted correlation curve in the correlation graph and prints the parameters.

Arguments

f	`scalar`	Line handle
g	`double`$[1 \times N]$	Auto- or cross-correlation
t	`double`$[1 \times N]$	Correlation delays [s]
T	`scalar`	Integration time [s]

Original file fitcorr.m

```
%Fit the correlation and extract the parameters.
%
% o      Line handle
% g      Correlation
% t      Delays [s]
% T      Integration time [s]
%
%If no input are given, the function prints the headings to the
%command window.
%
%Correlation function [1]:
%
%            1       1                1               Pt
% g(t) = 1 + - * {-------- * -------------------- + ------exp(-t/Tt)}
%            N    1 + t/Td   sqrt(1 + 1/S^2*t/Td)   1 - Pt
%
% [1] Widgengren J, Mets , Rigler R,
%     Fluorescence correlation spectroscopy of triplet states in solution:
%     A theoretical and experimental study,
%     J Phys Chem 99:13368-13379 (1995)
%
function fitcorr(o,g,t,T)
if ~nargin
   fprintf('\nFit model:    3D-Gaussian confocal volume, free diffusion, single component');
   fprintf('\nParameters:\n');
   fprintf('\n%14s%16s%16s%16s%18s\n\n','N','Td','Pt','Tt','zo/wo');
   return
end
try
   n=2e-7 < t & t < T/8;
```

```
    G=inline('(1./(1 + t./p(2))./sqrt(1 + t./(p(5)^2.*p(2))) + p(3)./(1 - p(3)).*exp(-t.*100./p(4)))./p(1)
       + p(6) - g','p','g','t');
    [p,R,r,e]=lsqnonlin(G,[1;1;0.1;2;5;1],[0.01;0.05;0.01;0.5;0.5;0.9],[250;50;0.8;5;100;1.1],optimset(
       'Display','off','MaxIter',1000,'MaxFunEvals',5000,'TolFun',1e-7,'TolX',1e-4),g(n),1e4*t(n));
    if e > 0
       set(o,'XData',t,'YData',G(p,0,1e4*t));
       fprintf('%16g%14gus%15g%%%14gus%16g\n',1e-3*round([1e3;1e5;1e5;1e3;1e3].*p(1:5)));
    else
       set(o,'XData',1,'YData',nan);
    end
catch
    set(o,'XData',1,'YData',nan);
end
```

Fit model [2]

$$G_{(\tau)} = G_\infty + \frac{1}{N}\left(\left(1+\frac{\tau}{\tau_d}\right)^{-1}\left(1+\frac{\tau}{S^2\tau_d}\right)^{-1/2} + \frac{P_t}{1-P_t}\exp\left(-\frac{\tau}{\tau_t}\right)\right)$$

Parameters

N	number of molecules
τ_d	diffusion time
$S = z_0^2/\omega_0^2$	elongation parameter
P_t	triplet fraction $\in (0, 1)$
τ_t	triplet lifetime
$G_\infty = \lim_{\tau\to\infty} G_{(\tau)}$	offset ≈ 1

where ω_0 is the confocal waist in the *xy*-plane and z_0 the confocal waist along the *z*-axis.

Assumptions

1. The confocal volume is approximatively a 3D-Gaussian given by

$$MDE_{(x,y,z)} = MDE_0 \exp\left(-\frac{x^2+y^2}{\omega_0^2}\right)\exp\left(-\frac{z^2}{z_0^2}\right)$$

2. The molecules diffuse freely and independently in 3D due to Brownian motion. The diffusion constant is related by

$$D \approx \frac{4\omega_0^2}{\tau_d}$$

5.3 Shutter control

A placeholder for automatic shutter control. Receives an open command when the user starts the correlator and a close command when the correlator is stopped.

Arguments

s	`scalar`	Shutter action {0:close, 1:open}

Outputs

s	`scalar`	Shutter state {0:closed, 1:open}

Original file shutter.m

```
%Control the shutter automatically.
%
function s=shutter(s)
```

6 Copyright

The MATLAB interface to the Flex99-12C USB hardware correlator is published as freeware. The author reserves the right to modify any of the contained files.

You are allowed to distribute this software as long as you deliver for free the entire, original archive. You are allowed to distribute modified versions of **export.m**, **fitcorr.m** and **shutter.m** as an extension package. In particular, you are not allowed to distribute a combination of modified files with other parts of the original package as a single archive.

Package contents Scripts for MATLAB R12 build 6.0.88 or newer

Path	File	Description
/	Flex99-12C.pdf	This user manual
	flex99.m	MATLAB interface
	Readme.txt	Summary
private/	export.m	Save the correlation data to a file
	fitcorr.m	Extract the parameters
	flex99c12.dll	MATLAB interface library
	flex99c12.m	MATLAB interface help
	shutter.m	Automatic shutter control

Package contents Executable binary for Windows 95/98/2000/XP

Path	File	Description
/	Flex99-12C.exe	User interface
	Flex99-12C.pdf	This user manual
	flex99c12.dll	MATLAB interface library
	Readme.txt	Summary

7 Warranty

Any warranty is strictly refused and you cannot anticipate any financial or technical support in case of malfunction or damage.

Feedback and comments are welcome. I will try to track reported problems and fix bugs.

References

[1] Bismuto E, Gratton E, Lamb DC, *Dynamics of ANS Binding to Tuna Apomyoglobin Measured with Fluorescence Correlation Spectroscopy,* Biophys J **81**: 3510–3521 (2001)

[2] Widgengren J, Mets Ü, Rigler R, *Fluorescence correlation spectroscopy of triplet states in solution: A theoretical and experimental study*, J Phys Chem **99**: 13368–13379 (1995)