User Manual
Agilent Acqiris
Streamer Analyzers

including

Base Test

and

Data Streaming Firmware

Models covered:

SC210 U1091ASC1

SC240 U1080A-002

Agilent Technologies

Manual Part Number

U1092-90007

Edition

A-RevI, February 2009

You can download the latest version of this manual from http://www.agilent.com/ by clicking on Manuals in the Technical Support section and then entering a model number. You can also visit our web site at http://www.agilent.com/find/acqiris. At Agilent we appreciate and encourage customer input. If you have a suggestion related to the content of this manual or the presentation of information, please contact your local Agilent Acqiris product line representative or the dedicated Agilent Acqiris Technical Support (ACQIRIS_SUPPORT@agilent.com).

Acqiris Product Line Information

USA (800) 829-4444

Asia - Pacific 61 3 9210 2890

Europe 41 (22) 884 32 90

CONTENTS

1. Out of the Box

1.1 Message to the User

Congratulations on having purchased an Agilent Technologies Acqiris data conversion product. Acqiris Digitizers/Analyzers are high-speed data acquisition modules designed for capturing high frequency electronic signals. To get the most out of the products we recommend that you read this manual carefully. We trust the product you have purchased will meet with your expectations and provide you with a high quality solution to your data conversion applications.

1.2 Using this Manual

The SC240 and SC210 Streamer Analyzer platforms are dual- and single-channel 6U CompactPCI®/PXI™ 1 GS/s per channel digitizers with on-board real-time data processing and high-speed optical data links.

The Analyzer functions are dependent on the firmware code loaded into the Data Processing Unit. They are described in this manual throughout the Firmware sections

This guide assumes you are familiar with the operation of a personal computer (PC) running a Windows 2000/XP or other supported operating system. It also assumes you have a basic understanding of the principles of data acquisition using either a waveform digitizer or a digital oscilloscope.

The manual is divided into 6 separate sections. To understand the elements of operation for the module it is essential that you carefully read chapters 1-5.

Chapter 1	***OUT OF THE BOX,*** describes what to do when you first receive your new Acqiris product. Special attention should be paid to sections on safety, packaging and product handling. Before installing your product please ensure that your system configuration matches or exceeds the requirements specified.
Chapter 2	***INSTALLATION***, covers elements of installation. Before attempting to use your Acqiris product for actual measurements we strongly recommend that you read all sections of this chapter.
Chapter 3	***PRODUCT DESCRIPTION***, provides a short description of all the functional elements of the SC210 and SC240 Modules.
Chapter 4	***FIRMWARE***, provides a detailed description of the Base Test and the Data Streaming firmware including algorithm and implementation aspects.
Chapter 5	***RUNNING THE ANALYZER DEMO*** Application describes the operation of the Analyzer Demo application. Analyzer Demo allows interactive operation of the SC240 / SC210 from a PC running a Windows 2000/XP operating system.
Chapter 6	***PROGRAMMING THE FIRMWARE,*** provides all the information required to write a software application based on the Data Streaming firmware.

For information necessary for writing your own software to control Acqiris products you should also refer to the **Programmer's Guide** and the **Programmer's Reference Manual**.

1.3 Conventions Used in This Manual

The following conventions are used in this manual:

⚠	This icon to the left of text warns that an important point must be observed.
WARNING	Denotes a warning, which advises you of precautions to take to avoid being electrically shocked.
CAUTION	Denotes a caution, which advises you of precautions to take to avoid electrical, mechanical, or operational damages.
NOTE	Denotes a note, which alerts you to important information.
Italic	text denotes a warning, caution, or note.
Bold Italic	text is used to emphasize an important point in the text or a note
`mono`	text is used for sections of code, programming examples, and operating system commands.

B,KB,MB,GB	is for Byte, KiloByte = 1024 bytes, MegaByte = 1024*1024 bytes, GigaByte = 1024*1024*1024 bytes
b,Kb,Mb	is for bit with multipliers as above.
`Triggered`	Denotes a VHDL object. It could be a single- or multi-bit signal or a component or a library name.
`0xn..n`	Denotes a hexadecimal value.

1.4 Model Names

Agilent Technologies Inc. acquired Acqiris SA and its product lines in December 2006. Use the tables below to cross reference the legacy model name and new Agilent numbers

Agilent Model Number	Acqiris Model Name
U1091ASC1	SC210
U1080A-002	SC240

1.5 Disclaimer and Safety

The SC Series CompactPCI Analyzer cards have been designed to operate in a CompactPCI/PXI crate. Power for the modules is provided by plugging them into a free slot (refer to the installation procedure). Agilent Technologies does not recommend operation of the SC210 or SC240 outside of a CompactPCI/PXI crate

⚠ *CAUTION*: ***Do not exceed the maximum input voltage rating! The maximum input voltage for 50 Ω input impedance is ±5 V.***

1.6 Warning Regarding Medical Use

The SC Series CompactPCI Analyzer cards are not designed with components and testing intended to ensure a level of reliability suitable for use in treatment and diagnosis of humans. Applications of these cards involving medical or clinical treatment can create a potential for accidental injury caused by product failure, or by errors on the part of the user. These cards are ***not*** intended to be a substitute for any form of established process or equipment used to monitor or safeguard human health and safety in medical treatment.

⚠ *WARNING*: ***The modules discussed in this manual have not been designed for making direct measurements on the human body. Users who connect an Acqiris module to a human body do so at their own risk.***

1.7 Packaging and Handling

Your Analyzer is shipped with the following components:

- A compact disc in an Agilent Technologies paper CD envelope that includes
  - 10 product User Manuals in electronic form (8-bit Digitizers, 10-bit Digitizers, 12-bit Digitizers, Averagers, Analyzers, Signal Analyzers, Streamer Analyzers, Time-to-Digital Converters, 3-, 5-, and 8-slot CompactPCI Crates, and the 21-slot CompactPCI Crate),
  - 1 Programmer's Guide and 1 Programmer's Reference Manual,
  - device drivers with sample software for different operating systems, environments and languages,
  - the Analyzer Demo application, a demonstration program for the AC/SC Analyzer products,
  - the AcqirisLive application, a demonstration program for our digitizer and averager products,
  - the SSR Demo application, a demonstration program for the Acqiris AP235/AP240 Analyzers,
  - the APX01 Demo application, a demonstration program for the Acqiris AP101/AP201 Analyzers,
  - the TC Demo application, a demonstration program for the Acqiris TC840/TC842/TC890 Time-to-Digital Converters,
  - full installation procedures for use with Microsoft Windows, National Instruments LabVIEW RT, Wind River VxWorks, IVI-COM/C, and Linux software.
- A declaration of conformity
- Optional documentation such as a model-dependent document giving Specifications & Characteristics, a Calibration Certificate, or a Performance Verification

After carefully unpacking all items, inspect each to ensure there are no signs of visible damage. Also check that all the components received match those listed on the enclosed packing list. Agilent cannot accept responsibility for missing items unless we are notified promptly of any discrepancies. If any items are found to be missing or are received in a damaged condition please contact the Agilent service center or your local supplier immediately. Retain the box and packing materials for possible inspection and/or reshipment.

1.8 Warranty

All Agilent Acqiris Digitizer products are warranted to operate within specification, assuming normal use, for a period of at least one year from the date of shipment. Units sold before April 2008 had three year warranties, as do some more recent ones; in case of doubt examine your invoice. It is recommended that yearly calibration be made in order to verify product performance. All repairs, replacement and spare parts are warranted for a period of 3 months. Warranty extensions are available as an option.

Agilent endeavors to provide leading edge technology that includes the latest concepts in hardware and software design. As such software and firmware used with the products is under continual refinement and improvement. All software and instrument firmware is supplied “as is” with no warranty of any kind. Software and firmware is thoroughly tested and thought to be functional at the time of shipment. At Agilent’s discretion software and firmware may be revised if a significant operational malfunction is detected.

In exercising this warranty, Agilent will repair or replace any product returned to the Agilent service center, within the warranty period. The warranty covers all defects that are a result of workmanship or materials. This excludes defects that are caused by accident, misuse, neglect, or abnormal operation.

The purchaser is responsible for returning the goods to the nearest Agilent service center. This includes transportation costs and insurance. Agilent will return all warranty repairs with transportation prepaid.

1.9 Warranty and Repair Return Procedure, Assistance, and Support

Agilent acquired Acqiris SA and its product lines in December 2006. Please contact your nearest Agilent Service Center before returning any product for repair.

You can find information about technical and professional services, product support, and equipment repair and service on the Web, see http://www.agilent.com/find/service (or http://www.agilent.com/ and after selecting your country click on **Contact Us**). The service center will ask for your name, company, phone number and address, the model and serial numbers of the unit to be repaired, and a brief description of the problem.

Before issuing a Service Order the service center may ask you to communicate with us by phone or eMail so that we can learn as much as needed about the problems observed. If a unit returned under guarantee is found to be working normally and this procedure was not followed we reserve the right to charge you for the work done.

For your nearest customer support center please contact Acqiris Technical Support (ACQIRIS_SUPPORT@agilent.com) or come visit our web site at http://www.agilent.com/find/acqiris. Alternatively, contact Acqiris at 1-800-829-4444 in the USA, +41 22 884 32 90 in Europe or +61 3 9210 2890 in the Asia-Pacific region. The Agilent Support Centers can also help redirect you for any questions concerning the installation and operation of your equipment.

1.10 System Requirements

Acqiris products need the following minimum PC System Requirements in order to obtain reasonable performance from the module:

Processor: 150 MHz Pentium (higher recommended). Some PowerPC systems running Wind River VxWorks are supported; please contact us for details.

Memory: 256 MB RAM. This number is a very rough estimate.

Display resolution: At least 1280 x 1024 pixels and 256 colors for use of the Analyzer Demo program is strongly recommended.

Operating System: Microsoft Windows 2000/XP including 2003 Server, Wind River VxWorks 5.5.1 and 6.4, and Linux with kernels 2.4 and 2.6. Support for Windows 95/98 is no longer available. Although this operating system is no longer supported by Microsoft, Windows NT4 users can download AcqirisSoftware 3.1 from the Agilent WEB site.

Hard Drive Space: 20 MB Minimum

CD Drive (or any method to copy the software installation files from CD to the hard drive such as LAN, floppy drive, etc.)

LabVIEW: Full driver implementations are available for National Instruments LabVIEW versions 8.5, 8.2.1, and 8.0. LabVIEW 7.1 is frozen at the level of Acqiris Software 3.2 with support for all instruments.

LabVIEW RT: National Instruments LabVIEW RT is supported for the same versions as shown above. The VISA driver must be version 3.0 or higher.

MATLAB: The MEX interface can be used with MathWorks MATLAB 7.3 or a newer version.

Visual BASIC: The interface files and examples are available for Microsoft Visual Basic .NET and Version 6.

Tornado: The example files are useable with Wind River Tornado 2.2.1

1.11 Transport & Shipping

***CAUTION:* Cards can be safely transported in their original shipping packages. DC cards can be transported when properly mounted in a CompactPCI crate. The transport of DP cards mounted in a PC is a more delicate issue. Because of their mass the cards can vibrate loose unless they are properly secured and braced. DP cards held only in the front and on the bottom should not be shipped in their PC. However, properly mounted DP cards with XP103 or XP105 fans can be sufficiently well held; the Adjustable retainer must be used so as to hold the card in place.**

To package the instrument for shipping:

Step	Notes
1. Place the instrument in its original packaging materials.	• If the original packaging materials are not available, use a professional packaging service. Contact your Agilent Service Center for more information.
2. Surround the instrument with at least 3 to 4 inches (8 to 10 cm) of its original packing material or bubble-pack to prevent the instrument from moving in its shipping container.	
3. After wrapping it with packing material, place the instrument in its original shipping container or a strong shipping container that is made of double-walled corrugated cardboard with 159 kg (350 lb) bursting strength.	• The shipping container must be large and strong enough to accommodate your instrument and allow at least 3 to 4 inches (8 to 10 cm) on all sides for packing material.
4. Seal the shipping container securely with strong nylon adhesive tape.	
5. Mark the shipping container “FRAGILE, HANDLE WITH CARE” to help ensure careful handling.	
6. Use the address obtained from your Agilent Technologies Service Center.	
7. Retain copies of all shipping papers.	

***CAUTION:* Damage can result if the original packaging materials are not used. Packaging materials should be anti-static and cushion the instrument on all sides. NEVER USE STYRENE PELLETS IN ANY SHAPE AS PACKAGING MATERIALS. They do not adequately cushion the instrument or prevent it from moving in the shipping container. Styrene pellets can also cause equipment damage by generating static electricity or by lodging in fan motors.**

1.12 Maintenance

The cards do not require any maintenance. There are no user serviceable parts inside. A periodic calibration can be obtained on request.

1.13 Cleaning

Cleaning procedures consist only of exterior cleaning.

Clean the exterior surfaces of the module with a dry lint-free cloth or a soft-bristle brush. If any dirt remains, wipe with a cloth moistened in a mild soap solution. Remove any soap residue by wiping with a cloth moistened with clear water. Do not use abrasive compounds on any parts.

1.14 Disposal and Recycling

Electronic equipment should be properly disposed of. Acqiris Digitizers and their accessories must not be thrown out as normal waste. Separate collection is appropriate and may be required by law.

2. Installation

This chapter describes how to install the Acqiris hardware and software for Windows 2000/XP, National Instruments LabVIEW RT, Linux, or Wind River VxWorks.

⚠ ***NOTE:*** *For a first time installation we strongly recommend installing the software* ***before*** *inserting the hardware into the PC.*

2.1 U1091AK02 IC414 Installation

⚠ ***NOTE:*** *If you are going to install an IC414 interface for the first time and are running Windows 2000/XP you should follow the procedure below* ***before*** *installing the Acqiris hardware.*

2.1.1 IC414 Hardware installation hints

The PCI-8570/PXI-8570 User's Manual (Rev. 1.00) section 2.5 gives Hardware Installation instructions.

⚠ *CAUTION*: ***Turn off the power of the PC; the PC may have to be unplugged to ensure that the PCI bus has no power available.*** Please ignore the PCI-8570 instruction to leave the power cord plugged in; Ground the chassis differently!

⚠ ***CAUTION:*** ***Touch the antistatic package to a grounded object before removing the card from the package. Electrostatic discharge can damage the card.***

The standard cable pair provided each have a red connector on one end and a black connector on the other. Therefore the correct connection can be made by plugging the Red connector into the L0Rx socket and the Black connector into the L0Tx socket on the PXI module and the other Red connector into the PCI module socket furthest from the PCI card internal base connector and the Black connector into the next socket.

If you intend to use 64-bit 66 MHz transfer to maximize data transfer speed you should cable a "bundled link" using two standard cable pairs and both the L0 and L1 pairs of connectors. You should also make sure that you configure the PXI-8570 M66EN Jumper correctly.

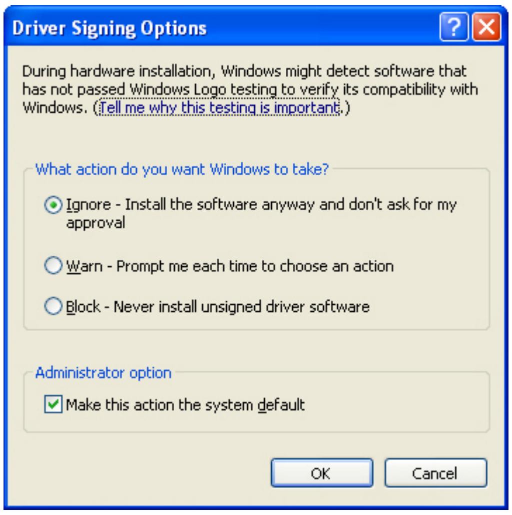
2.1.2 IC414 Windows software installation

Linux users do not need to read any further since there is no special software installation.

Windows users should have the hardware installed as noted above. This software installation should be done before any Acqiris modules are placed in the CompactPCI crates. This may mean that you have to remove the module from the crate *as delivered*.

The crate should be turned on first followed by the PC. If the cabling and start-up sequence is done correctly there will be no LED illuminated on the PCI unit connected pair and the LED's of the PXI connected pair will be lit.

For **Windows XP** installation, Select the **Control Panel** under Settings in the **Start** menu. Then, if you are using the **Category View** select **Printers and Other Hardware.** After this, for both **Category** and **Classic** views, go to **System** and then display the **Hardware** tab to get access to the **Driver Signing** menu. Since neither the AdLink nor the Acqiris driver has been submitted for **Windows Logo testing** you must select either the **Ignore** or **Warn** action. The resulting menu looks as shown:


Driver Signing Options

During hardware installation, Windows might detect software that has not passed Windows Logo testing to verify its compatibility with Windows. (Tell me why this testing is important.)

What action do you want Windows to take?

[✓ Ignore - Install the software anyway and don't ask for my approval]

Warn - Prompt me each time to choose an action

Block - Never install unsigned driver software

Administrator option

[✓ Make this action the system default]

The PCI-8570/PXI-8570 User's Manual (Rev. 1.00) section 2.4 contains the software installation instructions. These should be executed before allowing the hardware installation process to look for the driver. If you have an AdLink CD Version 2004A4 or later you can use it; if not you should download the latest driver from the WEB site (http://www.adlinktech.com/). You can then continue with the Hardware Installation. A reboot will then be necessary. At this point the Stargen Fabric PCI Adapter and the Stargen Aruba Fabric-to-PCI Bridge should appear correctly installed under System Devices in the Device Manager.

⚠ ***NOTE:*** If you have an AdLink CD Version 2005A3 or later you can find 8570install.exe in the folder X:\Driver Installation\PXI Platform\PXI Extension\PCI_PXI-8570\Wnt2kxp and the starfab1.inf file in the folder X:\Driver Installation\PXI Platform\PXI Extension\PCI_PXI-8570\Win98.

2.2 Installing the Software under Windows

2.2.1 Warnings

If Setup detects a previous installation of Acqiris software on your system, a warning screen will be displayed. It is recommended to exit Setup and uninstall older versions.

The installer from software releases prior to **Acqiris Software 2.0** installed the Digitizer Driver DLL files into the System directory. These will be removed by Setup. If you wish to keep the old installation on your system, you should exit Setup, and move all Acqiris driver files (acqiris*, acqrs* and acqir*) to some archive directory.

The DLL files will be installed into the bin subdirectory of the Acqiris software root, and the corresponding path will be added to the PATH environment variable.

2.2.2 Multiple Versions

With the software installation from **Acqiris Software 2.0** (or above), it is possible to keep multiple versions on the same system, but you must specify a different root directory (i.e. Install Folder). If you keep the same directory, Setup will overwrite your previous installation.

To go back to a previous version, you must change the PATH environment variable and reinstall the Kernel driver. Under Windows 2000/XP:

1. Copy the SYS file from `<old_AcqirisSoftware_root>\bin\kernel` to the `Windows\System32\drivers` directory.
2. Change the AcqirisDxRoot, AcqirisDxDir and PATH environment variables to the old root.
3. Reboot the computer.

2.2.3 Installation

Before installing the Acqiris hardware, you should complete the following steps to install the software for Windows 2000/XP.

⚠ ***NOTE:** You will need administrator privileges to complete the software installation under Windows 2000/XP.*

1. Insert the *Acqiris Software CD* into the CD-ROM drive of your computer. If the Autorun program does not start automatically (Autoplay disabled), you can start it manually, or navigate to the *AcqirisSoftware\Windows* folder in order to display the files included.
2. Choose **Install AcqirisSoftware for Windows 2000/XP/2003 Server** (or run `Setup.exe` from the *AcqirisSoftware\Windows* folder). After several seconds for initialization the first of many screens will appear.

Please note the following points:

- It is good practice to remove any previously installed version of Acqiris software. If the program finds that there is still Acqiris software installed on your machine a warning panel will appear.
- In the **Select Install Type** window selecting **Custom** installation will let you select individual packages for loading. The space indicated for LabVIEW, Firmware and UserManual packages is incorrect. The correct values are 7 MB, 40 MB, and 30 MB respectively. A full installation requires just under 150 MB of disk.
- If MATLAB is installed on your machine, you will be asked to point the installer to the MATLAB root directory. You should do this if you want the installer to modify the standard startup.m file to add the paths for the MEX interface.

- In the **Installation Folder** window you will give the name of the root directory of the Acqiris software installation. If User Manuals (30 MB) and Firmware (40 MB) are loaded more space than indicated here will be required on the drive. For the case of a Tornado 2.2 installation the folder name should not contain any spaces.
- AcqirisLive needs the LabWindows/CVI 8.0 Run-Time Engine to run. If Setup has detected that a LabWindows/CVI Run-Time Engine is already installed on your system, it will ask you if you would like to install it locally for AcqirisLive anyway. If you are not sure about the version of the CVI Run-Time Engine on your system, it is recommended to install it locally.
- The **Installation Summary** window will be shown to allow you to check what you have asked for. At this point it is not too late to go back and make changes. The actual installation will only be started after clicking on "Install" in the next window.
- Please read the **IMPORTANT Information** window text. It could help you avoid serious problems.
- Registration of your installation will help Agilent provide you with better support. You will also be notified of updates and upgrades. All information submitted will be treated in accordance with Agilent's privacy policy. Setup will prepare a registration e-mail in your e-mail client application upon termination of the setup procedure. You can then decide whether or not you wish to send it. You may also add comments. Uncheck the box if you do not want to register your installation.
- After the software installation is complete you can either accept the suggestion to restart the computer or you should shutdown your computer and proceed with the hardware installation.

2.3 Installing the Software for Linux

The Acqiris software is ready to install on Linux systems running kernels 2.4 and 2.6; package files are now available for some popular distributions. If you are using Debian, Ubuntu, Fedora, CentOS, or openSUSE, you can install the software using a standard package: see section 2.3.2 **Installing with a distribution specific package**.

2.3.1 Installing with a tar file

The following archives all contain the driver and library compiled with the **gcc** version shown:

`AcqirisLinux-3.3a-gcc-3.4.tar.gz` - compiled under CentOS 4.7 with GNU gcc 3.4.

`AcqirisLinux-3.3a-gcc-4.1.tar.gz` - compiled under Debian 'etch' 4.0 with GNU gcc 4.1.

The appropriate tar file should be copied to a local directory (e.g. your home directory) and then unpacked by using the following command

```
tar xzf AcqirisLinux-3.3a-gcc-X.x.tar.gz
```

The resulting directory **AcqirisLinux** contains an install script **drv-install** and a graphical Demo program **demo/AcqirisDemo**. Before running the software you have to install the acqrsPCI device driver into the running kernel

2.3.2 Installing with a distribution specific package

The distribution specific package files are located in directories named after the distribution name and version. Packages are available for the following variants:

- Debian, 4.0, 5.0 - the distribution used for the development of Acqiris software.

  Debian-4.0/agilent-acqirisdriver_3.3a-1_i386.deb

  Debian-5.0/agilent-acqirisdriver_3.3a-1_i386.deb
- Ubuntu, 8.04, 8.10 - this is the most popular and easy-to-use distribution, based on Debian.

  Ubuntu-8.04/agilent-acqirisdriver_3.3a-1_i386.deb

  Ubuntu-8.10/agilent-acqirisdriver_3.3a-1_i386.deb
- Fedora, 10 - another popular innovative distribution, test bed for RedHat Linux.

  Fedora-10/agilent-acqirisdriver-3.3a-1.i386.rpm
- CentOS, 4.7, 5.2 - this distribution is a rebranded build of RedHat Enterprise Linux.

  CentOS-4.7/agilent-acqirisdriver-3.3a-1.i386.rpm

  CentOS-5.2/agilent-acqirisdriver-3.3a-1.i386.rpm

- openSUSE, 11 - this distribution is the base system for Novell SUSE Linux.

  openSUSE-11/agilent-acqirisdriver-3.3a-1.i586.rpm

More information on these distributions is available on the following web-site:
http://distrowatch.com/dwres.php?resource=major

All these packages contain a version of the software Acqiris 3.3a release for Linux with full support of recent Acqiris products.

A deb package should be installed by the root user with the following command:

```
dpkg -i agilent-acqirisdriver_3.3a-1_i386.deb
```

An rpm package should be installed by the root user with the following command:

```
rpm -ivh agilent-acqirisdriver-3.3a-1.i386.rpm
```

Before being able the run the software, you have to install the acqrsPCI device driver into your running kernel as described in the next section.

2.3.3 Installing the Linux device driver

The **acqrsPCI** device driver is a loadable kernel module. It is provided as source files.

2.3.3.1 Compiling the kernel module

All files needed to compile a new kernel mode driver are in
src/agilent/acqiris/kernel26module/ for Linux kernels 2.6,
or src/agilent/acqiris/kernel24module/ for Linux kernels 2.4.

To compile the driver, the header files of the Linux kernel need to be installed. Furthermore, depending on the Linux distribution, the kernel source files may need to be installed too.

For Debian users, in order to compile the kernel module on a system, the kernel packages that match the current kernel-image must be installed. For instance, on a Debian ‘etch’ 4.0 system 686 the required packages are:

```
linux-image-2.6-686
linux-image-2.6.18-4-686
linux-headers-2.6-686
linux-headers-2.6.18-4
linux-headers-2.6.18-4-686
linux-kbuild-2.6.18
```

The makefile will try to guess where the kernel headers are located. If it fails, see instructions inside the makefile itself to specify manually where these kernel header files are located.

To compile a new kernel module, use the following commands:

```
cd kernel26module      (or kernel24module)
make clean all         to generate a new kernel module from scratch
make install           to copy the kernel module where it should reside
                        (this last command requires super user privileges).
```

In the end, the loadable kernel module should be present in the system's loadable kernel modules directory. You can verify it is present by running:

```
ls -l /lib/modules/$(uname -r)/extra/acqrsPCI.ko
```

Note that this version of the loadable kernel module had been tested on Linux kernel versions up to 2.6.27. However, starting with kernel 2.6.17, a few kernel functions relating to the device classes are no longer available to proprietary modules (such as this one). As such, automatic creation of the /dev/acqrsPCI node has to be done manually in the /etc/rc.local file (see **Creating the device driver node** in 2.3.4 **Loading the kernel module**).

2.3.4 Loading the kernel module

Once the device driver is installed into its kernel module directory, it has to be loaded into the kernel. First update the kernel modules dependencies:

```
/sbin/depmod -a
```

Then load the kernel module:

```
/sbin/modprobe -v acqrsPCI
```

You can add the command above to your `/etc/rc.local` file. It is run at system startup and will load the acqrsPCI kernel module automatically when your computer is restarted.

You can check that the device driver is loaded properly with lsmod.

```
/sbin/lsmod | grep acqrsPCI
```

If lsmod does not report a module named acqrsPCI, then the kernel module is not loaded into the kernel. Check any error messages that may have been issued by the commands you have run, and contact Agilent Acqiris technical support.

2.3.5 Creating the device driver node

Check that a device node has been created.

```
ls -l /dev/acqrsPCI
```

If the device node /dev/acqrsPCI is present on your system, the Acqiris Software is ready to be used. Otherwise, continue reading this section.

For systems using `udev` (the dynamic /dev/ hierarchy) or running kernels above 2.6.17, the device driver node may not be created automatically. You can do it with `mknod` by running the following command:

```
/sbin/mknod -m 666 /dev/acqrsPCI c 124 0
```

You can check that the device node has been created with `ls`:

```
ls -l /dev/acqrsPCI
```

You can add the `mknod` command to your /etc/rc.local file. It is run at system startup and will create the /dev/acqrsPCI device node automatically when your computer is restarted.

Note that the permissions that are set in this example (-m 666) give access to the Acqiris instruments to all users. This may not be what you expect and these permissions can be adjusted to fit your requirements. The base rule is that any user that wants to use the Acqiris instruments have to get read and write access to /dev/acqrsPCI.

2.3.6 Firmware files and Environment variables

If you intend to use Averagers, Analyzers, Time to Digital converters, or 12-bit digitizers you must install the Firmware .bit files. You can either copy them from the Firmware directory of the CD-ROM or download them from our WEB site. They should be placed in the Firmware subdirectory of AcqirisLinux. In the case of the CD-ROM, this can be done by continuing the above command sequence with the following:

```
cp /mnt/CDRom/Firmware/*.bit Firmware/
```

If you need to keep multiple versions of the Firmware, please continue reading this section.

Automatic loading of the firmware needed by some modules relies on the environment variable **AcqirisDxDir** pointing to the directory containing the file **AqDrv4.ini** which in turn points to the directory containing the Firmware **.bit** files. Therefore, assuming that your Firmware is in **/usr/local/AcqirisLinux/Firmware** and that **AqDrv4.ini** is in **/usr/local/AcqirisLinux/demo** then you must edit **AqDrv4.ini** so that it contains the line

fpgaPath=/usr/local/AcqirisLinux/Firmware

Then, if your shell is **csh** or **tcsh** modify the **.login** file to contain the line

setenv AcqirisDxDir /usr/local/AcqirisLinux/demo

or, if your shell is **bash, ksh, zsh** or **sh**, modify the **.profile** file to contain the lines

AcqirisDxDir=/usr/local/AcqirisLinux/demo

export AcqirisDxDir

2.4 Installing the Hardware

1. Turn off the power of the PC and the crate in the case of a CompactPCI module.

⚠ CAUTION: ***For PCI modules the PC may have to be unplugged to ensure that the PCI bus has no power available. However, CompactPCI crates can be left plugged in since this ensures proper grounding.***

⚠ CAUTION: ***Touch the antistatic package to a grounded object before removing the card from the package. Electrostatic discharge can damage the card.***

2. Module in a PC: open the PC, identify a free PCI slot and carefully insert the DP Series card into it. Be sure to ground yourself by touching the grounded PC frame and avoid touching any components on the DP Series card. Make sure that the grounding of the card's mounting bracket to the back panel rail of the computer is done correctly. If present make sure that the fan's adjustable retainer is correctly positioned and tightened for mechanical support. Close the PC.

   Module in a CompactPCI crate: Follow the instructions of the crate manufacturer to insert the DC Series card into a free 6U CompactPCI peripheral slot. Be sure to ground yourself by touching the grounded crate and avoid touching any components on the DC Series card. Be sure to tighten both front panel mounting screws to lock the module into place and insure proper grounding of the frame.

   ⚠ ***NOTE:*** *To ensure the best possible performance, users of Acqiris CC121 Crates with ASBus systems should respect the module placement rules to be found in the Agilent Acqiris 21-slot CompactPCI Crate User Manual.*

   PCI Bus extension module in a PC: Consult the manufacturer's documentation for any special instructions. Open the PC, identify a free PCI slot and carefully insert the card into it. Be sure to ground yourself by touching the grounded PC frame and avoid touching any components on the card. Make sure that the grounding of the card's mounting bracket to the back panel rail of the computer is done correctly. Close the PC. Connect the module to the CompactPCI crate controller.

3. Turn on the power of the crate(s), if present, and then the PC and start the operating system.

⚠ ***NOTE:*** *Acqiris modules are equipped with a LED. If this LED is not glowing orange or red when the power is applied there is a severe problem. Either the module is broken or the necessary voltages for its use are not available.*

⚠ ***NOTE:*** *For proper system operation when using the IC200, IC414, or other PCI extension interface to connect a CC10X crate to a remote PC, the crate* ***must*** *be powered on before the PC in order for the PC BIOS to recognize the presence of the CompactPCI crate.*

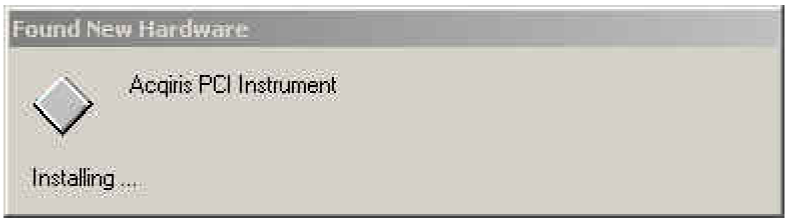
4. If devices were installed using a previous version of Acqiris software the instruments in these logical positions may still appear as *Unknown Devices*. This can be changed to the new Acqiris type category with the Grey Diamond icon by **Uninstall**ing the device and then **Installing** again. Instructions on this procedure can be found in the **ReadMe.txt** file in the **manuals** folder of your Acqiris software installation.

2.5 After Restarting

2.5.1 Windows 2000

Under Windows 2000, you *must* login with administrator privileges after the first boot following the hardware installation; the Plug&Play system must have the appropriate privileges to be able to complete your hardware installation successfully. After a successful hardware installation, you will be able to use your Acqiris module(s) with normal privileges.

At the first boot following the hardware installation, Windows will detect the new hardware and will install the devices automatically. The following image will appear.


Found New Hardware

Acqiris PCI Instrument

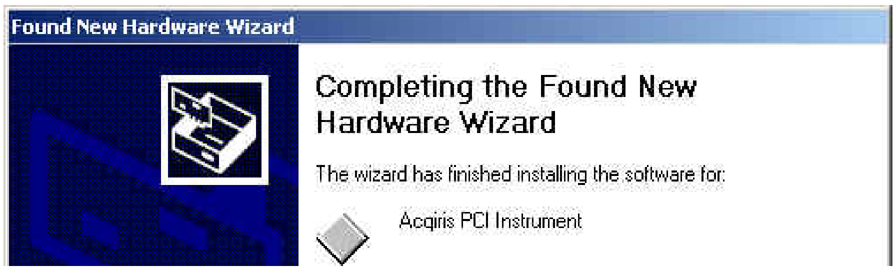
Installing ...

⚠ ***NOTE:*** *In some systems an application program (such as AcqirisLive) will not yet work correctly at this point. One additional boot cycle may be needed if this is the first time that a hardware board is being installed.*

2.5.2 Windows XP

Under Windows XP, you *must* login with administrator privileges after the first boot following the hardware installation; the Plug&Play system must have the appropriate privileges to be able to complete your hardware installation successfully. After a successful hardware installation, you will be able to use your Acqiris digitizer(s) with normal privileges.

If you login with administrator privileges after the first boot following the hardware installation, Windows will detect the new hardware and start the "Found New Hardware Wizard" after a few seconds. There is no need to use Windows Update to search for the software. You can "Install the software automatically". The final screen should appear as shown here:


Found New Hardware Wizard

Completing the Found New Hardware Wizard

The wizard has finished installing the software for:

Acqiris PCI Instrument

⚠ ***NOTE:*** *In some systems an application program (such as AcqirisLive) will not yet work correctly at this point. One additional boot cycle may be needed if this is the first time that a hardware board is being installed.*

2.6 LabVIEW RT

During program development you can choose whether you use LabVIEW or LabVIEW RT compatible libraries by switching the version present in **National Instruments\LabVIEW m.n\instr.lib\**. This swap can be facilitated by using the **Install VI library for LabVIEW or LabVIEW Real-Time** shortcut available in the Shortcut folder under **Start** → Programs.

There is only one Acqiris Driver. It supports all Acqiris Instruments. The instructions below concern LabVIEW RT as used in NI PXI processors.

The Aq_RT.inf and AqRT_4.ini files must be uploaded to the target. To do this,

- start the MAX application,
- right click on the target
- select file transfer
- select the Aq_RT.inf file on your host machine and upload ('To Remote') to the LabVIEW RT working directory (/NI-RT/system) on the target
- select the Aq_RT.ini file on your host machine and upload ('To Remote') to the LabVIEW RT working directory (/NI-RT/system) giving it the name AqDrv4.ini

For Acqiris modules which need FPGA files you should,

- create the folder \firmware in the /NI-RT/system directory using the file transfer application
- select the FPGA files (from <AcqirisDxRoot>\Firmware) you want to copy to the target and upload them into the firmware directory

Restart the target after finishing the file transfers.

Restart the MAX Explorer and you should have Acqiris digitizers detected in your PXI system.

2.7 Installing the IVI-COM/C Driver

Please install Acqiris software for Windows first. Then the *Acqiris Software CD* Autorun program gives access to two installers:

- IVI Shared Components 1.4
- IVI-COM/C IviAqD1 driver

These must be installed in the order shown above. For more information you can then consult the Readme.txt file in the IVI\Drivers\IviAqD1 folder or the documentation through the program shortcut present under Ivi/IviAqD1.

2.8 Distribution for Windows 2000/XP and Linux

The manuals/ReadMe.txt file contains a list of files to be found after a complete installation of Acqiris software on Windows systems. Similarly the ReadMeLinux file gives the list of files corresponding to that installation.

3. Product Description

3.1 Overview

The SC Series are powerful analyzer platforms based on a single-channel (SC210) or dual channel (SC240) 1 GS/s per channel, 8-bit CompactPCI digitizer with on-board real-time data processing implemented with the aid of a large field-programmable gate array (FPGA). In addition, they provide fast data transfer via optical fibers, at rates of up to 30 Gbits/s (depending on option).

The SC Series Architecture offers a convenient way to implement user defined data processing algorithms with data rates of up to 2 GS/s. It provides external processing control through dedicated connectors and visual reference by means of two front-panel LEDs.

Key Features

- **Dual-Channel Performance with Interleave -** The SC240 offers dual-channel synchronicity for I/Q acquisitions with up to 1 GS/s sample rate. Interleaved single-channel mode up to 2 GS/s on either input is software selectable. The SC210 provides a single input channel with a sample rate of up to 1 GS/s.
- **On-board reconfigurable Data Processing Unit (DPU) –** The SC Series analyzer platform can easily be reconfigured to perform a variety of user-defined on-board real-time signal processing tasks on the digitized signals. The on-board FPGA is capable of executing multiplications in less than 5 ns and offers more than 74,000 logic cells, up to 7 Mbits of on-chip RAM, and 328 dedicated 18-bit x 18-bit multipliers with 36-bit results.
- **Optional external Processing Memory** – As an option, the SC Series can be equipped with additional memory to extend the processing capability of the DPU. The P512M Memory option offers two independent banks of 256 MB of DDR333-SDRAM and a 1 MB dual-port SRAM**.**
- **High-Speed Optical Fiber Links** – The SC Series provides very high speed data transfer capabilities through front-panel connectors. The standard version offers two 2.5 Gbit/s optical fiber links, for an aggregate useful unidirectional throughput of up to 480 MB/s. Optionally, the SC analyzers can be equipped with twelve 2.5 Gbit/s optical links, for an aggregate useful unidirectional throughput of up to 2.88 GB/s.
- **Front-panel I/O Connectors and Controls** for real-time data processing control (DPU Ctrl^2) – It provides several front-panel connectors for real-time control of the DPU. The function of these connectors and LEDs is user defined through the implemented firmware.

3.1.1 Coupling & Impedance

Each channel has a 50 Ω signal input BNC connector giving high quality termination with better than ± 1% precision. It is ideally suited for use with 50 Ω transmission lines. Both AC and DC coupling modes are available. The AC mode couples signals capacitively thus removing the input signal's DC component and filtering out any signal component below 32 Hz. The DC mode allows all signal components to be passed through to the digitizer.

3.1.2 Input Protection

The input amplifiers are fully protected against over-voltage signals. Input signals up to ±5 V DC at 50 Ω, can be input without damaging the front-end electronics.

3.1.3 Mezzanine Front-end

The front-end electronics are all mounted on a removable mezzanine card. In the event of accidental damage, or as components fatigue over time (e.g. relays in high duty cycle automated testing applications), the mezzanine card allows for fast and efficient replacement.

3.1.4 Bandwidth and Rise Time

The bandwidth specification indicates the frequency at which an input signal will be attenuated by 3 dB (approximately 30% loss of amplitude). The bandwidth also affects the minimum rise and fall times that can be passed through the front-end electronics. A pulse with a very sharp edge will be observed to have a minimum rise time (τ_{min}) determined by the front-end electronics. In general a pulse with a given 10-90% rise time ($\tau_{10\text{-}90real}$) will be observed with a slower value given by:

$\tau_{10\text{-}90}^2 = \tau_{10\text{-}90real}^2 + \tau_{min}^2$ where τ_{min} (ns) ≈ 0.35 (GHz-ns) / BW (GHz)

If desired, a hardware bandwidth limiter at 20, 200 or 700 MHz can be selected.

Model Agilent #	**Bandwidth into 50 Ω**	**Minimum Rise Time**
SC210, **SC240** ≈ U1080A	1GHz	0.35 ns

3.1.5 Input Voltage and Offset

The input channel provides a fully programmable amplifier with variable input voltage and offset. Full Scale (FS) input voltages are selectable from 50 mV to 5 V in a 1, 2, 5 sequence. Care should be taken to select an input voltage range that will allow the signal to be recorded using as much dynamic range of the digitizer as possible. The variable offset is programmable in the range of ± 2 V, in the FS ranges 50 to 500 mV, and ± 20 V, in the FS ranges 1 V to 5 V. The raw 8 bit ADC data values are in the range [-128,+127] with the first and last values reserved for underflow and overflow respectively. The midpoint value, 0, of the range corresponds to the negative of the offset voltage. Thus the Full Scale Range (FSR) goes from

–Offset Voltage **– (FS/2)** to –Offset Voltage **+ (FS/2)**

Signals going outside of the FSR will be clipped and data values for the clipped portion of a signal should be regarded as erroneous.

3.1.6 Vertical Resolution

The SC Series Streamer Analyzers use an ADC system with 8 bits of vertical resolution (256 levels). The dynamic range of the ADC covers the Full Scale (FS) of the Input Voltage setting. For example, if the Input Voltage is set to 500 mV then the ADC resolution is equivalent to 1.95 mV. The SC Series Analyzers use low noise front-end electronics in order to ensure a good signal to noise ratio. To obtain the best dynamic range from the ADC care should be taken to ensure that the input signal varies over more than 50% of the Input Voltage Full Scale (FS) setting. The highest and lowest levels of the ADC correspond to underflow and overflow conditions.

3.1.7 DC Accuracy

DC voltage accuracy is better than ± 2% (± 1% typical) of the input voltage full-scale. The differential linearity is better than ± 0.9 LSB

3.2 Trigger

In normal digitizer operation, the trigger signal stops the data acquisition, typically somewhat delayed, depending on the user-specified *delay* value.

When the card is operated as a continuous data analyzer, the trigger signal is routed to the DPU where it is used as an indicator of the time region of interest. However, it does not stop the acquisition.

3.2.1 Trigger Source

The trigger source can be a signal applied to either the Input Channel (for internal triggering) or the External Trigger Input.

The modules provide a front panel BNC External Trigger Input. The External Input provides a fully functional trigger circuit with a fixed 50 Ω termination impedance, as well as selectable level and slope. It has the same BW limiter selections as can be found for the input channel. The user can choose the external trigger Full Scale from the set of values 0.5, 1.0, 2.0 or 5.0 V. A ±5 V limit on trigger signals should be respected, although somewhat higher voltages for short time periods will not damage the unit.

3.2.2 Trigger Coupling

Trigger coupling is used to select the coupling mode applied to the input of the trigger circuitry. Modes available include AC LF Reject, HF Reject, and DC. The AC LF Reject mode couples signals capacitively and removes the input signal's DC component and frequencies below 50 Hz. DC mode allows all signal components to be passed through to the trigger circuit. The HF Reject mode removes signal components above 50 kHz.

3.2.3 Trigger Level

The trigger level specifies the voltage at which the selected trigger source will produce a valid trigger. The trigger level is defined as a set voltage. Using the internal trigger with DC coupling, the level is set with respect to the midpoint voltage (V_m= – Offset voltage) of the digitizer's vertical scale. Internal trigger level settings must be within $V_m \pm 0.5$ FS, where FS is the channel or External Full Scale. All trigger circuits have sensitivity levels that must be exceeded in order for reliable triggering to occur. Trigger levels are also adjustable when using AC coupling.

SC Series digitizers will trigger on signals with a peak-peak amplitude > 15% FS from DC to their bandwidth limit.

3.2.4 Trigger Slope

The trigger slope defines the direction of the signal that will be used to initiate the acquisition when it passes through the specified trigger level. Positive slope indicates that the signal is transitioning from a lower voltage to a higher voltage. Negative slope indicates the signal is transitioning from a higher voltage to a lower voltage.

3.2.5 Window Trigger

The SC Series digitizers implement a Window trigger. Two trigger level thresholds are used to define the desired range. The trigger can then be chosen to occur either when the signal exits or enters the window range. This mode can be thought of as the appropriate OR of two edge triggers of opposite slope.

3.2.6 HF Trigger

The SC Series digitizers implement an HF trigger that allows triggers to be reliably accepted at rates above ~ 1 GHz. In this mode, triggers occur on every fourth positive edge. The window trigger is not available in this mode.

3.2.7 Pre- and Post-Trigger Delay

The pre- and post-trigger counters are only used in normal digitizer mode. When the card is operated as a continuous data analyzer, no acquisition trigger is required. The pre- and post-trigger counters are therefore ignored at this level.

To increase trigger flexibility a pre- or post-trigger delay can be applied to the trigger position.

The amount of pre-trigger delay can be adjusted between 0 and 100% of the acquisition time window (i.e. sampling interval x number of samples), whereas the post-trigger delay can be adjusted between 0 and 200 million samples.

Pre- or post-trigger delays are just different aspects of the same trigger positioning parameter:

- The condition of 100% pre-trigger indicates that all data points are acquired prior to the trigger, i.e. the trigger point is at the **end** of the acquired waveform.
- The condition of 0% pre-trigger (which is identical to a post-trigger of 0) indicates that all data points are acquired immediately after the trigger, i.e. the trigger point is at the **beginning** of the acquired waveform.
- The condition of a non-zero post-trigger delay indicates that the data points are acquired after the trigger occurs, at a time that corresponds to the post-trigger delay, i.e. the trigger point is **before** the acquired waveform.

The digitizer hardware accepts pre- and post-trigger adjustments in increments of 16 samples. By definition post-trigger settings are a positive number and pre-trigger settings are a negative number.

Thus it is only natural that the software drivers provided treat pre- and post-trigger delays as a single parameter in seconds that can vary between *–nbrSamples * samplingInterval* (100% pre-trigger) and *+maxPostTrigSamples * samplingInterval* (max post-trigger of 200M samples). Since the Acqiris software drivers provide very accurate trigger position information upon waveform readout, the accepted resolution of the user-requested pre-/post-trigger delay is much better than 16 samples. For more details, refer to the **Programmer's Reference Manual.**

3.2.8 Trigger Status

The front panel includes a tri-color LED indicator to show the status of the trigger for normal digitizer mode. This LED indicator is located near the TRIGGER IN BNC Connector.

When the LED is green it indicates the trigger is armed and waiting for a valid trigger to occur. Red indicates that the trigger has occurred, the acquisition is complete, and the data is waiting to be read out. The user can override the default functions and program the LED color in an application-specific manner.

When the SC Series are used as an analyzer this LED is always green when the acquisition is continuously running.

3.3 Sampling Rate

All Acqiris digitizers contain an analog-to-digital conversion (ADC) system that can sample waveforms, in a real time sampling mode, at rates from the maximum allowed rate down to 100 S/s (10 ms per point). The sampling rate can be programmed and is selectable in a 1, 2, 2.5, 4, 5 sequence (i.e. 1 MS/s, 2 MS/s, 2.5 MS/s, 4 MS/s, 5 MS/s, 10 MS/s, etc.). The maximum sampling rate shown above sometimes exploits the possibility of combining channels. The SC210 can sample up to 1 GS/s and the SC240 up to 2 GS/s. The data of all of the active channels is acquired synchronously; all of the ADC's are acquiring data at the same time, to within a small fraction of the maximum sampling rate.

3.4 Data Acquisition - Digitizer Mode

Data from the ADC are stored in on-board acquisition memory. The amount of memory in use for acquisition can be programmed and is selectable from 2 points to 128 Kpoints, the full amount of acquisition memory available.

For technical reasons, a certain memory "overhead" is required for each waveform, reducing the available memory by a small amount. In order to simplify programming, an interface function recommends the best sampling rate and the maximum possible number of data points, taking into account the available memory, the requested time window, the number of segments (in Sequence mode), as well as the required memory overhead.

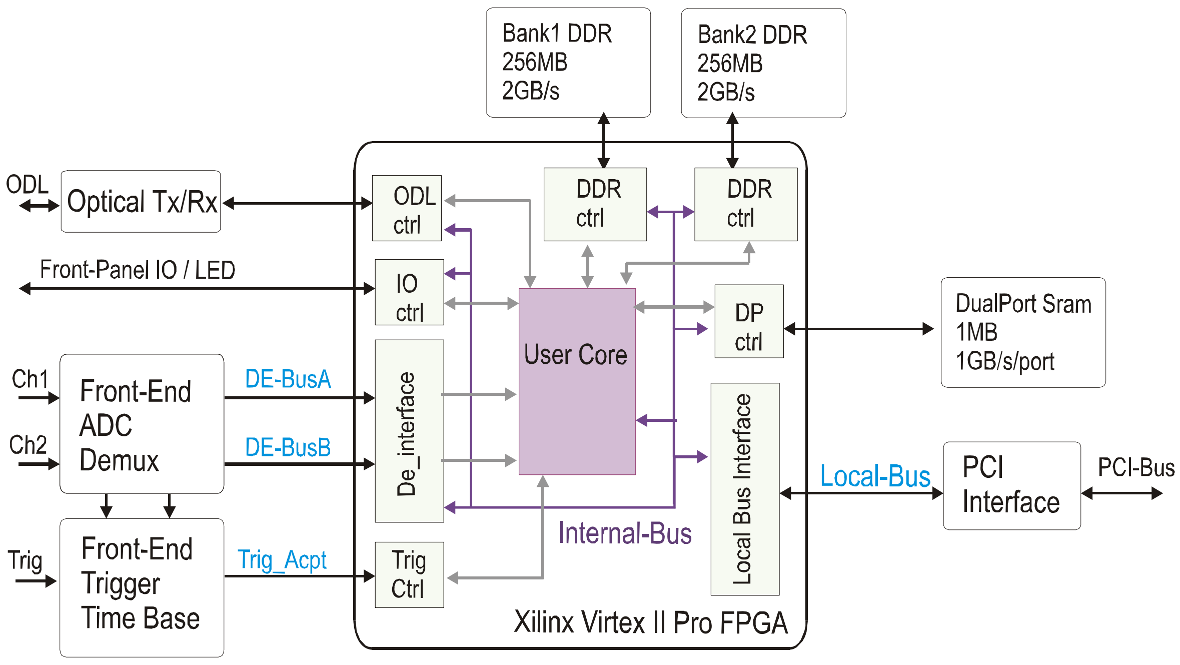
The Time Base Range defines the time period over which data is being acquired. For example, the SC210 in the digitizer mode has an acquisition memory of just under 128 Kpoints and maximum sampling rate of 1 GS/s. Thus, at the maximum sampling rate, the digitizer can record a signal over a time period of up to 130 µs (128 Kpoints * 1 ns/point). The time base range can be adjusted by varying the amount of acquisition memory or the sampling rate of the digitizer.

3.5 Data Processing – Analyzer Mode

Data from the ADC are continuously streamed through the data demultiplexer (MAC) to the on-board Data Processing Unit. The Data Processing Unit of the SC Series is implemented as a large field-programmable gate array (FPGA) with optional external memory.

The figure below presents the Data Processing hardware environment and some of the basic blocks used to interface them.

The SC Series are equipped with 2 to 12 bidirectional Optical Data Link running at up to 2.5 Gbits/s.



3.5.1 Data Processing Unit

The Data Processing Unit is well suited for many data processing schemes. It is implemented as a powerfull FPGA, the Xilinx Virtex II Pro XC2VP70-6FF1517, that provides up to two embedded PowerPCs, 328 18-bit x 18-bit multipliers, and 328 block RAMs.

The main features are summarized in the table below. Please refer to http://www.xilinx.com for the latest information about this device.

Resources	Qty	Description / Comment
Logic cells	66176	1 Logic cell has 1x (4 Input LUT + Flip-Flop + Carry Logic)
Block RAM	328	Instances of block RAM, 18 kbits each
Multiplier	328	18-bit x18-bit multipliers
DCM	8	A digital clock manager including frequency synthesis and phase shifting features. Frequency up to 420 MHz.
BUFG	16	Global Clock Buffer
Rocket IO	16	2.5 Gbit/s serial transceivers, of which 12 instances are simultaneously usable in the design
PowerPC	2	No support from Acqiris

In addition, the P512MB Memory option can improve the processing capability.

The Analyzer functions are completely dependent on the firmware downloaded into the Data Processing Unit. The application specific firmware available for the SC2x0 Series is described in chapter 4 ***FIRMWARE***.

As an option, a FDK (Firmware Development Kit) for the Acqiris SC2x0 series enables users to develop and integrate user-specific data processing algorithms. The FDK reduces the development effort by providing a set of cores that interface to the underlying hardware resources. Please refer to the FDK Reference Manual for more information.

3.5.2 Memory Option

The external DPU memory option consists of:

- 2 banks of 256MB of DDR333 SDRAM with a throughput of up to 2 GB/s per bank.
- 1 MB of dual-port SRAM with a read/write throughput of up to 1 GB/s per port.

Each DDR bank is built around four 512Mbit DDR-SDRAM devices that are organized as 8M x 16bits x 4 banks. Data are synchronously transferred on a 64-bit wide bus on each edge of a 166.67 MHz clock. A DDR Controller that is able to sustain continuous burst reading or writing at a data rate of up to 2 GB/s is provided as part of the FDK.

The Dual Port Memory is built around two 4Mbit Dual Port SRAM organized as 128K x 36 bits. Data are synchronously transferred on a 64-bit wide bus on each port using a clock at up to 166.67 MHz. A Dual Port Controller that is able to sustain continuous simultaneous burst reading and writing at a data rate of up to 1 GB/s is provided as part of the FDK.

3.5.3 Optical Data Links

The SC Series provides two Optical Transceivers to extend the Data Processing Unit towards external Processing or Storage Units. There are two options, based on two different types of optical transceivers.

By default the SC Series is equipped with the Standard(2) Data Rate link that consists of 2 Small Form Pluggable transceivers, providing two bidirectional optical data links at up 2.5 Gbit/s each.

The HRODL(12) *Hi Rate Optical Data Link* option contains 2 Parallel Optic transceivers that handle up to 12 optical data links at 2.5 Gbit/s in each direction.

The Optical Data Link bandwidth depends both on the bit rate and on the link protocol efficiency. The SC Series implements a local clock synthesizer that can generate the following bit rates: 1.0625 Gbit/s, 2.125 Gbit/s, and 2.5 Gbit/s.

A Link Layer core that is compatible with the Serial Front Panel Data Port standard (sFPDP) is part of the Firmware Development Kit.

The Optical Data Link physical layer depends both on the RocketIO© Multi Gigabit transceivers of the Data Processing Unit and on the transceiver characteristics.

Standard Data Rate

Parameter	Value	Comments
Connector Form Factor	Duplex LC	
Bit Rate	Up 2.5 Gbit/s	1.0625 Gbit/s and 2.125 Gbit/s also available (☝)
Number of Channels	2	1 RX / 1 TX per transceiver
Optical Wavelength	850 nm	Short Wavelength SFP (☝)
Link Length	150 m (max)	50 µm Multimode Fiber

(☝) Contact Acqiris for other values

Hi-Rate Optical Data Link

Parameter	Value	Comments
Connector Form Factor	MTP© (MPO12-F)	Use U1091ACB2 (XO100) for connecting to LC targets.
Bit Rate	Up 2.5 Gbit/s	1.0625 Gbit/s and 2.125 Gbit/s also available (☝)
Number of Channels	12	12 RX or 12 TX per transceiver
Optical Wavelength	850 nm	Short Wavelength SFP (☝)
Link Length	100m (max)	50 µm Multimode Fiber

(☝) Contact Acqiris for other values

⚠ ***CAUTION*: *LASER Radiation: Do not view directly with optical Instruments. Class 1M LASER Products***

⚠ ***CAUTION*: *The optical port plug provided should be installed anytime a fiber cable is not connected.***

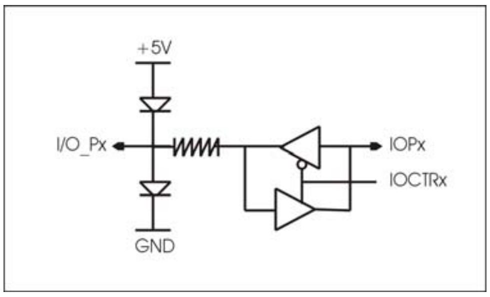
3.5.4 Extended Data Processing Controls

The SC Series provides several front-panel connectors for real-time control of the DPU. The function of these connectors and LEDs is user defined through the implemented firmware.

Two front-panel digital I/O MMCX-type connectors (I/O P1 & P2) are dedicated to the direct control of the data processing unit. These signals are 3.3 V compatible CMOS.

Each digital I/O can independently be configured either as an input or as an output. The figure below shows the equivalent schematic of one I/O Px interface.

The series resistor value is 50 Ω. I/O Px and I/O CTRx signals are connected to the Data Processing Unit.

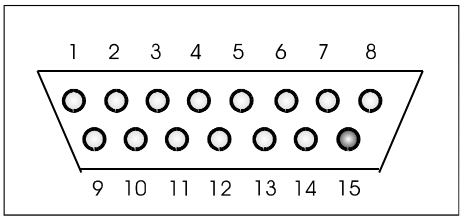


A third MMCX front-panel coaxial connector (ANL Out) is an analog output signal whose voltage is driven by a 16-bit on-board serial DAC. This analog signal can be used in simple control systems. The voltage range of that signal is –5V to +5V. The typical settling time (full scale range) is 1µs.

A front-panel µDB-15 connector (I/O EXT) provides fourteen bi-directional direct lines to the DPU that can be used as seven differential pairs or as fourteen closely coupled single ended lines. Please note that these lines must use 2.5V signaling logic standards.

⚠ **CAUTION**: ***Do not exceed the maximum input voltage rating! The maximum input voltage for µDB-15 Connector is 2.6 V.***

The figure below shows the pinout of the µDB-15 connector. Pin 15 is connected to the electrical ground.



The table below gives the pinout allocation of the IO_EXT connector. Each DPxn/DPxp pair refers to two lines routed towards the data processing unit as a differential pair.

Pin	Allocation
1 - 2	DP6n – DP6p
3 - 4	DP5p – DP5n
5 - 6	DP4p – DP4n
7 - 8	DP0p – DP0n
9-10	DP3p – DP3n
11-12	DP2p – DP2n
13-14	DP1p – DP1n
15	GND

Finally, two LEDs (L1 & L2) provide a visual reference. Each LED is independently driven by the Data Processing Unit and displays one of the four following colors (Black or switched off, Red, Green, or Orange).

3.6 External Clock and Reference

For applications where the user wants to replace the internal clock of the digitizer in order to drive the ADC with an external source, an External Clock or Reference signal input is available. The Clock or Reference signals can be entered into the digitizer via the MMCX CLK IN connector on the front panel.

When using an External Clock, the user must ensure that the input signal has a frequency between 10 MHz and 2 GHz, and a minimum amplitude of at least 1 V, peak to peak, into 50 Ω. The External Clock allows the digitizer to make a voltage measurement whenever the clock signal passes through a predefined threshold. However, it should be noted that when 2 channels are being used the maximum Sampling Rate is half of the External Clock Frequency in the Continuous mode and in this case the Start/Stop mode is to be preferred. The threshold range is variable and user selectable between ± 2 V. The signals should not exceed ±5 V amplitude.

For applications that require greater timing precision and stability than is obtainable from the internal clock, a 10 MHz Reference signal can be used. The amplitude and threshold conditions, for an External Reference, are the same as for the External Clock. If phase synchronization between several digitizers is required, the reference signal should be applied to all of them.

3.7 Internal Calibration

The software drivers supplied include calibration functions for the timing, gain, and offset settings, which can be executed upon user request. The digitizers are never calibrated in an "automatic" way, i.e. as a side effect of another operation. This ensures programmers have full control of all calibrations performed through software in order to maintain proper event synchronization within automated test applications.

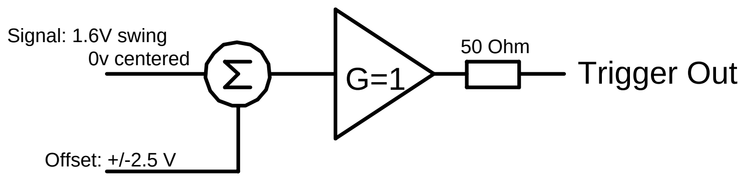
The SC2x0 includes a high precision voltage source and a 16-bit DAC, used to determine the input voltage and offset calibration.

For accurate time and voltage measurements it is recommended to perform a calibration once the module has attained a stable operating temperature (usually reached with a few minutes after power on). Further calibration should not be necessary unless temperature variations occur. Calibration can usually be performed with signals present at the channel, external, and clock inputs. However, if the calibration is found to be unreliable, as shown by a calibration failure status, it may be necessary to remove such signals.

3.8 External Trigger Output

When the module is ready to be triggered and a valid trigger signal occurs, a trigger output is generated for external use. It is always available on the Front Panel Trigger Out MMCX connector. The pulse ends when the data acquisition for the trigger in question is complete.

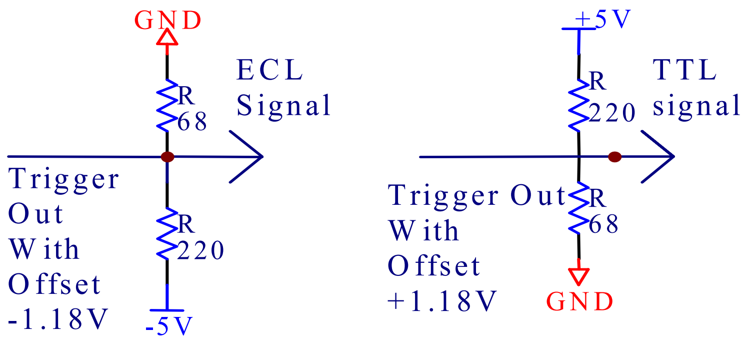
Trigger Output Block diagram:



The output swing is 1.6 V (± 0.8 V) when unloaded and 0.8 V when terminated on 50 Ω. The rise and fall times are 2.5 ns typical. The offset can be adjusted, by software control in the range [–2.5 V, +2.5 V] unloaded, or [-1.25 V, +1.25 V] into 50 Ω. The maximum output current capability is ± 15 mA. As the output is retro-terminated, it is possible to drive a 50 Ω line unterminated (HiZ) without loss of performance.

For a TTL compatible signal, set the offset to 1.0 V and the swing at destination will be +0.2 to +1.8 V.

For an ECL compatible signal, terminated on 50 Ω to –1.2 V, set the offset to –1.2 V and the output will be in the range [–0.8 V, –1.6 V]).



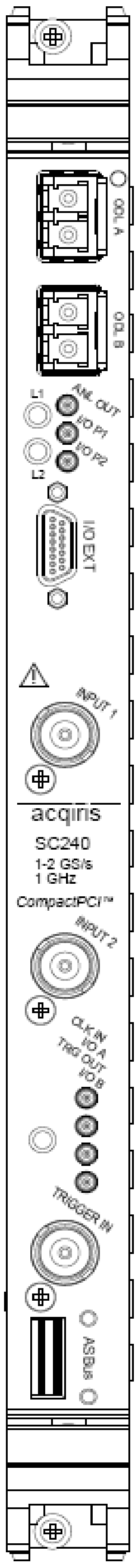
Alternatively, to reduce the current drawn from the digitizer, the terminations shown here can be used:

The trigger out signal can also be routed to the PXI Bus Star Trigger line.

3.9 SC240/SC210 Front Panel Inputs and Controls

Name	Generic Function	Comments	Connector
ODLA	Optical Data Link 1	Duplex LC (Tx/Rx)	ODL A
ODLB	Optical Data Link 2	Duplex LC (Tx/Rx)	ODL B
L1	LED Status (*)	Firmware Dependent	LED L1
L2	LED Status (*)	Firmware Dependent	LED L2
ANL Out	Analog Output (*)	Firmware Dependent	MMCX ANL OUT
I/O P1	Input/ Output (*)	Firmware Dependent	MMCX I/O P1
I/O P2	Input/ Output (*)	Firmware Dependent	MMCX I/O P2
I/O Ext	7 differential lines for remote control or 14 closely coupled single ended lines (*)	Firmware Dependent	I/O EXT
INPUT 1	Signal input	Signal input (Channel 1)	BNC INPUT 1
INPUT 2	Signal input	Signal input (Channel 0)	BNC INPUT 2
CLK IN	Reference clock	50 Ω Input for external clocking	MMCX TR
I/O A	User configurable	–	MMCX I/O A
TRIGOUT	Signal occurs after an accepted TRIGGER. It is synchronous to the acquisition Clock and can be used to trigger events synchronously to the acquisition clock.	Also available when using the SC Series Module as a standard digitizer.	MMCX TRIG OUT
I/O B	User configurable	–	MMCX I/O B
TRIGGER IN	Trigger input	Trigger Input	BNC TRIGGER IN
AS bus	Auto Synchronous Bus System	–	AS bus
Digitizer Status LED	Acquisition Status	Green when data is streamed to DPU	LED next to TRIG OUT

(*) See 3.5.4 ***EXTENDED DATA PROCESSING CONTROLS***

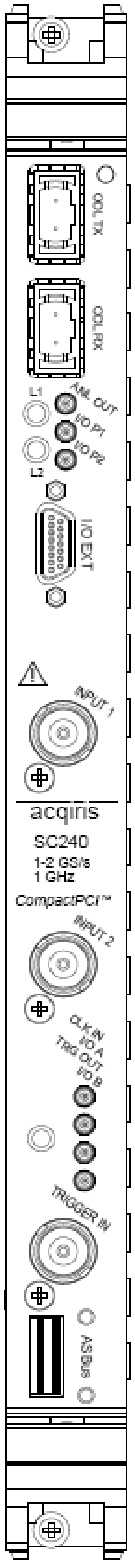


The I/O A, I/O B signals are 3.3 V compatible CMOS. This means that, on input, low must be < 0.7 V and high must be in the range [1.7 V, 5.0 V]. An unconnected signal will be high. This definition ensures TTL compatibility. On output, the low level will be in the range [0 V, 0.7 V] and the high level in the range [1.7 V, 3.3 V] for HiZ. The high level output will typically generate 0.8 V into 50 Ω.

For firmware dependent inputs and controls, please refer to the corresponding sections within the firmware description section.

NOTE The SC210 Front Panel is similar, with the exception that there is only one Signal Input named INPUT and located in place of the INPUT2 BNC.

3.10 SC240/SC210-HRODL(12) Front Panel Inputs and Controls



Name	Generic Function	Comments	Connector
ODL TX	TX Optical Data Link 1-12	MPOF-12 Connector	ODL A
ODL RX	RX Optical Data Link 1-12	MPOF-12 Connector	ODL B
L1	LED Status (*)	Firmware Dependent	LED L1
L2	LED Status (*)	Firmware Dependent	LED L2
ANL Out	Analog Output (*)	Firmware Dependent	MMCX ANL OUT
I/O P1	Input/ Output (*)	Firmware Dependent	MMCX I/O P1
I/O P2	Input/ Output (*)	Firmware Dependent	MMCX I/O P2
I/O Ext	7 differential lines for remote control or 14 closely coupled single ended lines (*)	Firmware Dependent	I/O EXT
INPUT 1	Signal input	Signal input (Channel 1)	BNC INPUT 1
INPUT 2	Signal input	Signal input (Channel 0)	BNC INPUT 2
CLK IN	Reference clock	Input for external clocking	MMCX TR
I/O A	User configurable	-	MMCX I/O A
TRIGOUT	Signal occurs after an accepted TRIGGER. It is synchronous to the acquisition Clock and can be used to trigger events synchronously to the acquisition clock.	Also available when using the SC Series Module as a standard digitizer.	MMCX TRIG OUT
I/O B	User configurable	-	MMCX I/O B
TRIGGER IN	Trigger input	Trigger Input	BNC TRIGGER IN
AS bus	Auto Synchronous Bus System	-	AS bus
Digitizer Status LED	Acquisition Status	Green when data is streamed to DPU	LED next to TRIG OUT

(*) See 3.5.4 ***EXTENDED DATA PROCESSING CONTROLS***

The I/O A, I/O B signals are 3.3 V compatible CMOS. This means that, on input, low must be < 0.7 V and high must be in the range [1.7 V, 5.0 V]. An unconnected signal will be high. This definition ensures TTL compatibility. On output, the low level will be in the range [0 V, 0.7 V] and the high level in the range [1.7 V, 3.3 V] for HiZ. The high level output will typically generate 0.8 V into 50 Ω.

For firmware dependent inputs and controls, please refer to the corresponding sections within the firmware description section.

NOTE The SC210 Front Panel is similar with the exception that there is only one Signal Input named INPUT located in place of the INPUT2 BNC.

3.11 Physical Specifications

3.11.1 Electrical

SC210 / SC240		CURRENT REQUIREMENTS (A)			
Firmware	**Max. Power Consumption (W)**	**+12V**	**+5V**	**+3.3V**	**-12 V**
Base Test	22.3	0.7	0.8	2.2	0.05
Standard(2) Data Streaming	29.5	0.8	1.15	3.3	0.05
HRODL(12) Data Streaming	36.9	0.8	1.18	5.3	0.05

The Maximum Power Consumption has been increased by 10% over the value calculated with the currents shown to take into account higher allowed values of the crate or PCI voltages.

These modules use the PCI Bus at 33 MHz and are compatible for either V I/O = 3.3 V or 5 V. All of these modules are capable of DMA transfers at rates ~100 MB/s.

3.11.2 Environmental and Physical

Operating Temperature

0° to 40°C

The above values are for the ambient temperature of the room (or equivalent) where the SC210/SC240 is located. The temperature as measured on the board may well be significantly higher. On-board temperatures above 60°C should be avoided.

Relative Humidity

5 to 95% (non-condensing)

Dimensions

All SC modules conform to the CompactPCI standard and have a 6U form factor. (233 mm × 160 mm × 20 mm).

Safety

Complies with EN61010-1

EMC Immunity

Complies with EN61326-1: Industrial Environment

EMC Emissions

Complies with EN61326-1: Class A for radiated emissions

Required Airflow

> 2 m/s in situ

4. Firmware

The following sections describe the major elements of firmware supplied by Acqiris, as standard or as an option.

The term ‘firmware’ refers to the FPGA program. It is contained in a file with the extension ‘.bit’ that must be downloaded by the software driver through the Compact PCI bus to configure the SC240/SC210 Module for a specific operating mode.

Once configured and started through the Compact PCI Bus, the firmware typically needs little or no interaction with the controlling PC, nor further support from the software driver, until explicitly stopped with a software command. In some cases, the user may continuously monitor the data streaming process by capturing some data blocks on the fly, without lowering the data throughput.

The table below presents the list of the available firmware for the SC Series.

Firmware Name	Description	Firmware file	Order Information
Base TEST Firmware - Default FPGA configuration files for initial loading and diagnostic tests	Base Test for SC210	SC210.bit	Standard
	Base Test for SC240	SC240.bit	Standard
Base TEST Firmware - Supports up to 1 link for a theoretical aggregate transfer rate of 240 MB/s	1ODL / 2 Channel	SC240Str1.bit	Standard
Base STREAMING Firmware - Supports Standard(2), up to 2 links for a theoretical aggregate transfer rate of 480 MB/s	2 ODL / 1 Channel	SC210stream2.bit	Standard
	2 ODL / 2 Channels	SC240Stream2.bit	Standard
Base STREAMING Firmware - Supports HRODL(12), up to 12 links for a theoretical aggregate transfer rate of 2.8 GB/s.	12 ODL / 2 Channels	SC240Stream12.bit	Standard

NOTE: Both the Base Test and Data Streaming firmware are always delivered with an SC module. The VHDL source code is only provided for the BASE Firmware with the FDK option. Source code for the DATA STREAMING Firmware is not available.

NOTE: For the Data Streaming firmware there is a practical limit of 2 GB/s because data cannot be generated any faster.

Others firmware may be protected by permission codes which prevents their use on unauthorized SC2x0 modules. Protected firmware can only be used on SC2x0 modules that contain the corresponding permission code for the firmware option.

4.1 Base Test Firmware

The Base Test firmware is designed to run on any Acqiris AC2x0 or SC2x0 Analyzer platform card. The Base Test firmware is built from a base design database whose purpose is twofold:

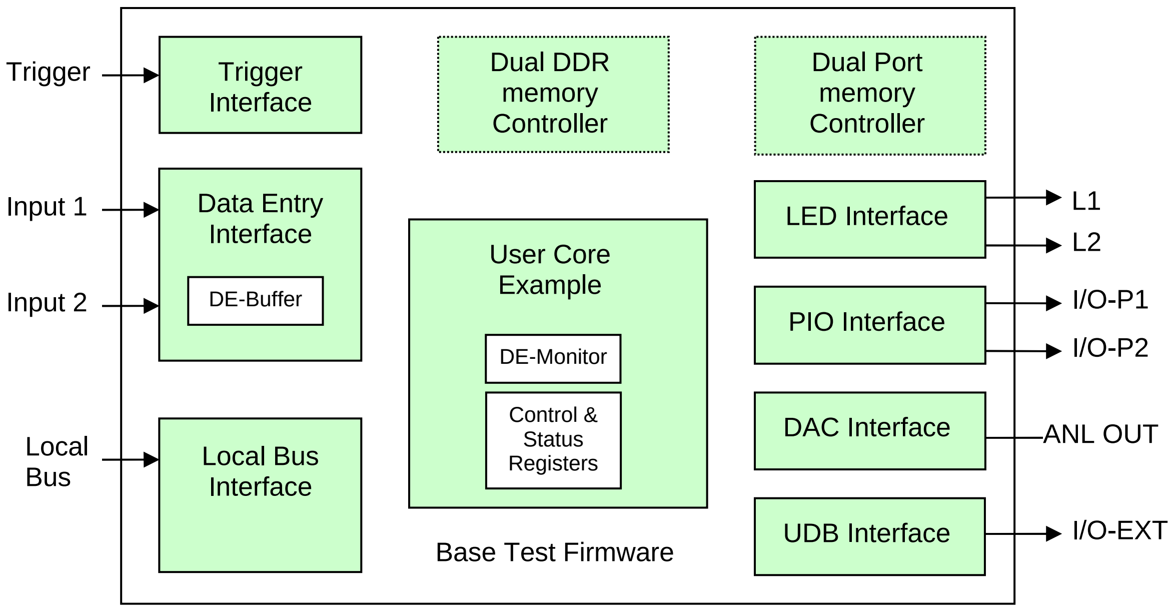
- “Design framework”. It is meant to be used as a starting point for any new developments based on the FDK
- Test and Demonstration. It could be used for test/demonstration purposes of either the SC2x0 underlying hardware or the developed firmware itself.

There is one base test for the SC210 module and another one for the SC240 module.

Users of the Firmware Development Kit (FDK) should, as part of the FDK installation and verification, rebuild this firmware. It is included in source form in the FDK option.

4.1.1 Architecture of Base Test Firmware

The following picture presents the simplified architecture of the Base Test firmware for the SC240 Module.



The Base Design is a collection of Agilent Acqiris-supplied cores (filled in green on the block diagram) that interface with the underlying hardware of the module.

The *Local Bus Interface* core enables the communication between FPGA registers and buffers and the board local bus connected to the PCI bridge. The *DE interface* core provides an interface to the data entry bus to retrieve the demultiplexed data samples. It contains an 8K sample FIFO, named DE buffer, which can be read back through the *Local Bus* interface. The *Trigger interface* core is used to place the trigger information within the incoming data stream. The *LED Interface* core controls the color of both L1 and L2 LEDs. The *PIO interface* core is used to manage both I/O-P1 and I/O-P2 MMCX. The *DAC interface* core provides control of the 16-bit DAC connected to the ANL-OUT MMCX. The *UDB Interface* core provides a way to test the µDB connector. Finally, the *Dual DDR Controller* core and *Dual Port Controller* core can be used to store and read samples or data of interest, provided that the P512MB option is implemented on the board.

The Base Design contains a *User Core* example that implements two registers and one 8K sample buffer named DE-Monitor Buffer. It provides a minimum set of functions to reduce the developer’s work when starting a new design and to perform tests on the hardware resources. While using the Analyzer Demo application, the user can capture a waveform on both channels at two levels (in the DE buffers and in the DE-Monitor Buffers) and access the various resources (e.g. LED, PIO, DAC, etc…) through the core registers. In addition, it implements some basic tests to check the health of the SC module.

4.1.2 I/O and Controls of Base Test Firmware

The table below presents the functionality of the I/O controls located on the front panel.

Item	Generic Function	Comments	Connector
L1	LED Status	Software LED	LED L1
L2	LED Status	DE-Monitor Buffer Full	LED L2
ANL Out	Analog Output	Driven by DAC core	MMCX ANL OUT
I/O P1	Input/ Output	Driven by FPIO core / Used for Debug	MMCX I/O P1
I/O P2	Input/ Output	Driven by FPIO core / Used for Debug	MMCX I/O P2
I/O Ext	7 differential lines for remote control or 14 closely coupled single ended lines (*)	Used to Test uDB link provided specific loop hardware is used.	I/O EXT

The LED colors can be overwritten by the software using the FPGA front-panel LED control register. The ANL-Out value can only be driven by using the FPGA front-panel DAC control register. I/O-P1 and I/O-P2 values are configured as outputs and can multiplex several signals of interest using the FPGA front panel PIO control register. Finally, the I/O Ext connector is only used for production tests with this firmware.

4.2 Base Streaming Firmware

The Base Streaming firmware is designed to run on an Acqiris SC240 Streamer Analyzer platform card. Its purpose is to be the starting point for customers developing specialized streaming applications with the FDK option. Its function and optical link management is made very simple and easy to understand.

Key Features

- **High Speed –** The Data Streaming firmware provides a useful throughput of up to 240 MB/s on a single optical fiber link.
- **Simple frame –** The data frames are simple frames with only a header of 128 bits.
- **Standard serial FPDP Transfer Protocol –** The Base Streaming firmware uses the standard serial FPDP protocol, making the data stream compatible with a number of commercially available disk storage systems that are especially designed for high-speed data storage.
- **Concurrent Monitoring –** While the data streaming is in operation, the user can observe the data being transmitted. The monitoring is concurrent with the normal operation, and does not influence it at all.
- **High Resolution Trigger** – The resolution of the built-in trigger core is 1 ns.
- **Front Panel Indicators** – In order to ease the integration and synchronization of the Data Streaming firmware in a large variety of systems, two front panel LEDs show critical information on current status.

4.3 Data Streaming Firmware

The Data Streaming firmware is designed to run on an Acqiris SC240 or SC210 Streamer Analyzer platform card. Its purpose is to stream digitized data continuously from the analog-to-digital converter across one or several optical serial links to data storage systems or processing units. The source code of this firmware is not part of the FDK option.

Key Features

- **High Speed –** The Data Streaming firmware provides very high speed data transfer capabilities. The Standard(2) version supports two 2.5 Gbit/s optical fiber links, for an aggregate useful throughput of up to 480 MB/s. On instruments with HRODL(12), the optional 12-link PAROLI2 transceiver, the firmware supports up to twelve 2.5 Gbit/s optical fiber links, for an aggregate useful data streaming throughput of 2 GB/s.
- **Standard serial FPDP Transfer Protocol –** The Data Streaming firmware uses the standard serial FPDP protocol, making the data stream compatible with a number of commercially available disk storage systems that are especially designed for high-speed data storage.
- **Concurrent Monitoring –** While the data streaming is in operation, the user can observe a complete data block on demand. The monitoring is concurrent with the normal operation, and does not influence it at all.
- **Front Panel Indicators** – In order to ease the integration and synchronization of the Data Streaming firmware in a large variety of systems, two front panel LEDs show critical information on current status.

4.3.1 Data Streaming Mode

There are three operating modes:

1. In **Continuous** mode, the streaming starts as soon as streaming is enabled by the user program, independently of any hardware trigger synchronization. The firmware will continuously stream the samples, channel 1 and/or channel 2, to the transmission buffer as configured with the user program. The sample rate shall not exceed the total maximum link throughput.
2. In **Start On Trigger** mode, the streaming starts at the first trigger occurrence after streaming is enabled by the user program and then continues to stream the samples to the transmission buffer as configured with the user program. The sample rate shall not exceed the total maximum link throughput.
3. In **Triggered** mode, while the framing process on the optical link is still continuous, the data stream is not continuous. Once streaming is enabled by the user program, after each trigger a programmable number of samples will be sent to the transmission buffer as configured with the user program. While the sample rate may be set to the maximum value, the average throughput, trigger to trigger shall not exceed the total maximum link throughput.

NOTE: The triggered mode is not available for the sc210 firmware of the **Standard(2) streamer** and not available for the sc210 and sc240 **HRODL(12) streamer**.

4.3.2 Architecture of Data Streaming Firmware

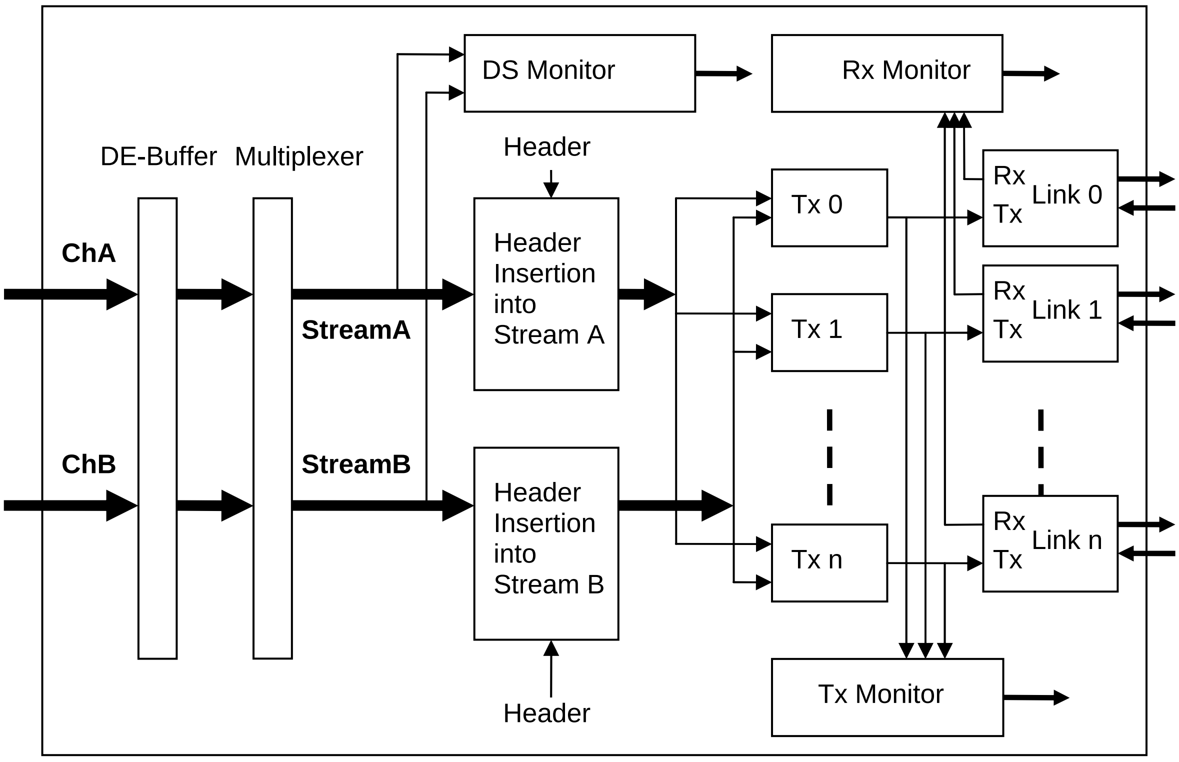
The analog input signals are passed through signal-conditioning amplifiers, where the coupling, offset, and gain can be programmed. In the SC210, and in non-interleaved operation of the SC240, each signal is sampled at up to 1 GS/s and converted to 8-bit values. They are demultiplexed to blocks of 16 samples at up to 62.5 MHz, and passed to the data processing FPGA. In interleaved operation of the SC240, a single signal (Input 1 or 2) is converted by both ADCs in a time-shifted manner, so as to effectively achieve twice the conversion rate of a single ADC.

The drawing below shows the data flow inside the Data Streaming firmware. The converted data from the ADCs are transmitted as data flows, ChA and ChB, to the data processing FPGA (DPU) where they are received by a data entry buffer (DE Buffer). The data are channeled through a multiplexer to the required internal data stream A or B.

In case of an SC240 module, the front panel input **INPUT1** corresponds to the data flow B and the front panel input **INPUT2** corresponds to the data flow A. In case of an SC210, the front panel input **INPUT1** correspond to the data flow A.

The data streams are partitioned into *Stripe Frames* of programmable length (see next section). If a header was requested, the header is inserted at the beginning of each stripe frame. Finally, the stripe frames are distributed in a 'round-robin' fashion to the appropriate transmission buffers Tx. The association between data streams and the data links is programmable.

The implementation of the optical link controller enables monitoring the frames on the receiver side. This feature is implemented for demonstration and test purposes. While the Tx Link is connected back to the Rx link, we can correlate and verify that received data are strictly identical to the transmitted data.



4.3.3 Distribution of a Stream to Multiple Data Links

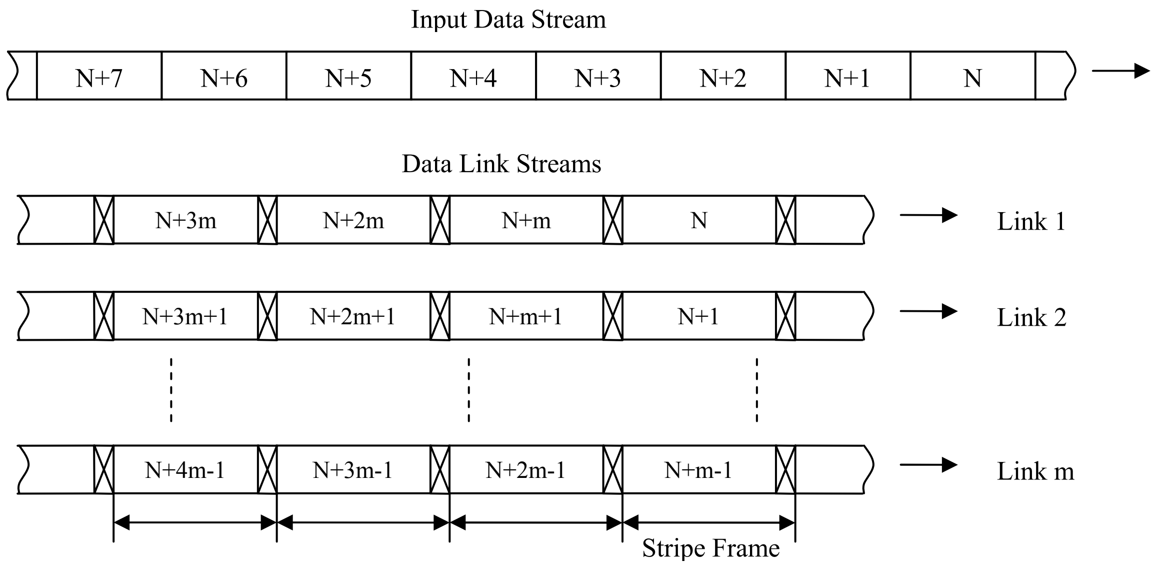
The Streamer Firmware is designed to transmit data from up to two different input channels to up to twelve output data links.

Each input channel can be configured to generate a data stream of up to 1G/s using either an internal or external sampling clock reference. Each optical data link can be used to serialize data at a rate of up to 240MB/s, using the *Serial Front Panel Data Port* (sFPDP) Protocol.

The data are continuously sent over the user-assigned data links. Obviously the data transmission will be lossless **only** if the input data flow rate is less than the total available transfer bandwidth on the output data links.

4.3.4 Data Structure

The continuous data flow of each stream is split into consecutive data blocks of ***equal*** size. They are inserted into **Stripe Frames** that consist of a fixed size header and the fixed size data block. Stripe frames are transmitted in a 'round-robin' fashion to all available data links, as illustrated below.



The stripe frame header may be 0, 16, or 32 bytes long.

The fixed size data block (stripe frame size) is ***fixed*** during a recording session. However, it may be modified for different recording sessions, since it is programmable in steps of 16 bytes. Its value is recorded in the stripe frame header, unless the header is omitted.

The maximum stripe frame size is 16 KB, including the header, while the minimum is 128 Bytes. From the standpoint of transfer efficiency, the stripe frame size should be chosen as large as possible.

NOTE In the case of 2 interleaved ADCs, typically at 2 GS/s, when both ADCs of an SC240 are combined into a single (logical) data stream, each physical data stream is treated separately. Thus, all even samples are transferred into Data Stream A and distributed to its associated links, while all odd samples are sent to Data Stream B and distributed to other links. When reassembling the recorded data into a single logical stream, the 2 different stripe frames must be identified (same *streamID* !) and their data must be interleaved into a single array of consecutive data.

In the SC240, there are always 2 data streams active, whether the digitizer is in 2-channel mode or in single-channel interleaved mode. The 2 data streams, A and B, run synchronously, i.e. they form 2 parallel stripe frames that begin and end at the same time. The 2 parallel stripe frames have the same *Stripe Frame Number* and are distinguished by different *streamID* values, see the next section for more details.

4.3.5 Stripe Frame Structure

A Stripe Frame contains two fields: an optional header field and the data payload.

The size of a Stripe Frame is programmable through the *Stripe Frame Size* field (not including the header size) of the Streamer Configuration register. The unit of this field is the *Sample Block* that consists of 16 samples. Values ranging from 8 up to 1024 are allowed, equivalent to sizes of 128B to 16 KB.

The Header field can be used to identify and ease data retrieval at the Data Storage side. The Header Type is a configuration parameter that is defined by the field **`HTYP`** of the **`Streamer Global Configuration Register`**. This field defines the type of the header field that will be appended in front of each Stripe Frame.

HTYP	Header Mode	Header Size
00	No Header	0 Bytes
01	Short Header (without Synchronization Marker)	16 Bytes
10	Long Header (with Synchronization Marker)	32 Bytes
11	Reserved	N/A

The following table describes the format of the Long Header, in 32-bit words:

	31 - 28	27 - 24	23 - 22	21 - 20	19 - 16	15 - 0
Word 0	HTYP	Link #	Reserved	StreamID	Reserved	Stripe Frame Size (bytes)
Word 1	Stripe Frame Number					
Word 2	Time Stamp Lo					
Word 3	Empty		Time Stamp Hi			
Word 4	Synchronization Marker : SoftSync Bytes 0 - 3					
Word 5	Synchronization Marker : SoftSync Bytes 4 - 7					
Word 6	Synchronization Marker : SoftSync Bytes 8 - 11					
Word 7	Synchronization Marker : SoftSync Bytes 12 - 15					

Name	**Comments**
HTYP	Header Type
Link #	Link Number that was used to transmit this Stripe Frame. (0 to 11)
StreamID	Stream Identifier (0 = Stream A, 1 = Stream B)
Stripe Frame	Number of bytes in Stripe Frame, **without** the

Name	**Comments**
Size	header bytes. Is constant throughout a recording.
Stripe Frame Number	Unique monotonously increasing frame number, starting from 0 when the recording is started.

1. The Time Stamp marks (in units of data samples) the position of the beginning of the current stripe frame, with respect to an arbitrary starting position, typically the beginning of the recording.
2. The 16 bytes of SoftSync values are a programmable synchronization pattern that is constant throughout a recording. The Streamer Soft Sync registers define this pattern. In the case of data loss, there is a high probability that the next Stripe frame could be identified by advancing through the data stream until this fixed pattern is encountered.

4.3.6 Output Bandwidth

The Streamer firmware can handle simultaneously up to two data streams at 1GS/s as produced by data acquisition on both input channels with a 1GS/s sampling rate (SC240).

The available output data bandwidth is provided by 2 or 12 optical data links (depending on the underlying hardware option). Each optical data link is controlled by a serial link controller that implements the Serial FPDP interface capable of sustained data throughput of up to 240MB/s.

Each link can be allocated to one of the two available data streams or left unused. The **Streamer Global Configuration Register** defines the association between one output link and one data stream. This register also contains the parameters that are common to both data streams (Header Type, Streaming Mode).

Once the framing process is started, the Data Streaming Firmware uses the contents of the **Streamer Global Configuration Register** to define an allocation table that is used in a round robin manner. The framing process is performed whatever the output bandwidth available. Each time a stripe frame is completed, the data streaming firmware will look at the next entry of that table to determine whether or not it can transmit the stripe frame. If the incoming data bandwidth is greater than the available output data bandwidth, the next data link might not yet be available. In this case, the current stripe frame is overwritten by a new stripe frame from the continuous input data flow. Data loss is then flagged into the **Streamer Status Register**. It can also be observed at the receiver end by looking at the Stripe Frame Number field that is embedded in the Header. This number is incremented even if the stripe frame is skipped due to insufficient output bandwidth.

4.3.7 Monitoring

The Data Streaming firmware contains monitoring buffers on the internal data streams (DS Monitor) as well as on the transmission (TX-Monitor) and reception lines (RX-Monitor) of the data links. They can be configured to capture a stripe frame at the requested point during operation without interfering with it in any way. After the capture, the monitoring buffers can be read by the host computer, again without disturbing the ongoing streaming process.

4.3.8 I/O and Controls of Data Streaming Firmware

The table below shows the functionality of the I/O controls located on the front panel.

Item	**Generic Function**	**Comments**	**Connector**
L1	LED Status	Stream A Status : • Green: At least one ODL successfully initialized and/or active in Tx Mode. • Red: At least one ODL faulty or all ODL not initialized. • Orange: Streaming mode initialized without ODL errors	LED L1
L2	LED Status	Stream B Status : • Green: At least one ODL successfully initialized and/or active in Tx Mode. • Red: At least one ODL faulty or all ODL not initialized. • Orange: Streaming mode initialized without ODL errors	LED L2

ANL Out	Analog Output	Not Used	MMCX ANL OUT
I/O P1	Input/ Output	Driven by FPIO core / Used for Debug	MMCX I/O P1
I/O P2	Input/ Output	Driven by FPIO core / Used for Debug	MMCX I/O P2
I/O Ext	7 differential lines for remote control or 14 closely coupled single ended lines (*)	Used to Test uDB link, provided specific loop hardware is used.	I/O EXT

I/O-P1 and I/O-P2 values are configured as outputs and can multiplex several signals of interest using the FPGA front panel PIO control register. For this firmware the I/O Ext connector is only used for production test.

5. Running the Analyzer Demo Application

The **Analyzer Demo** application is an interactive program running under Windows. It controls the operation of some Agilent Acqiris-supplied applications on the AC2x0 Analyzers and on the SCx0 Streamer Analyzers.

5.1 Getting Started with Analyzer Demo

Once the software and hardware installation described in Section 2 of this manual is complete, you can start **Analyzer Demo** from the start menu of your computer.

During startup, **Analyzer Demo** searches for all Acqiris Digitizers/Analyzers on the PCI/CompactPCI bus. If none are found, **Analyzer Demo** will display an error message indicating this fact and automatically switch to simulation mode with three different simulated instruments. In such a case the solution may be to turn off the computer, install and turn ON the hardware on the CompactPCI bus, and then restart the computer.

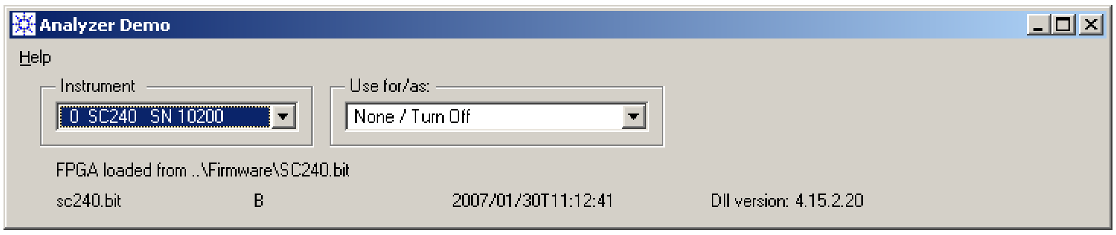
When you start **Analyzer Demo**, it displays three independent windows, an instrument control window, a digitizer control window, and an application window. If several instruments are present, there may be an application window for each of them.

5.2 Instrument Control Window

The instrument control window is common to all available instruments. The drop-down list on the left-hand side allows the selection of the *current* instrument, if several are available.

The drop-down list on the right-hand side "Use for/as" allows choosing an application for the *current* instrument. The application window below automatically opens/adapts and the appropriate FPGA configuration file is automatically loaded. On SC2x0 Streamer Analyzers, at least the ***Base Test*** and a ***Streamer*** application are present, as described further below. Other applications might be available, depending on the options installed in the card.

The digitizer control window always acts on the *current* instrument, see next section.



Instrument: 0 SC240 SN 10200
Use for/as: None / Turn Off
FPGA loaded from ..\Firmware\SC240.bit
sc240.bit B 2007/01/30T11:12:41 Dll version: 4.15.2.20

At the bottom of the instrument control window, the name of the loaded FPGA configuration file is shown, together with its directory path. A few additional pieces of information, such as the firmware name and version, are also shown (subject to change).

5.3 Digitizer Control Window

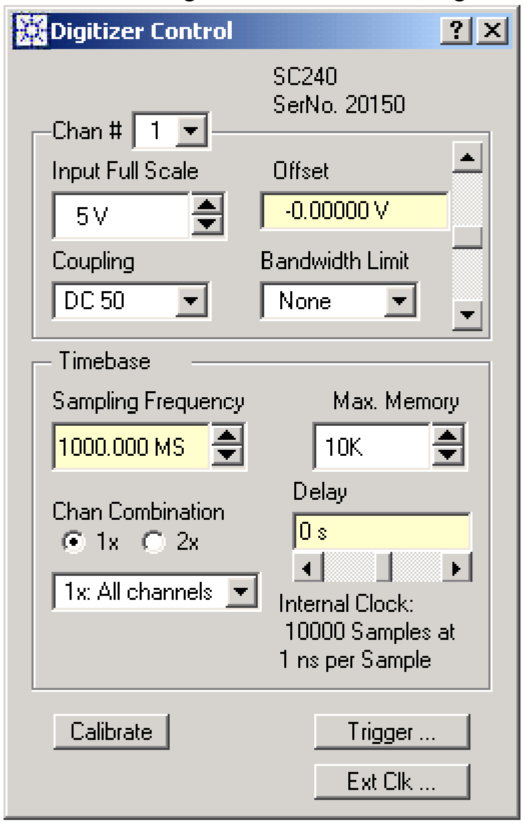
The Digitizer Control panel provides all the features needed to use the SC240/210 module in the Digitizer mode. It allows setting up the appropriate instrument configuration for the signal(s) applied to the front panel input(s), e.g. input coupling, full scale, and sampling rate.

The Digitizer Control panel always acts on the currently selected instrument in the instrument control window.

The model and serial numbers of the currently controlled instrument are shown at the top right of the control window.

If several channels are available in the instrument, the **Chan #** field permits which channel is currently being displayed and/or modified.

The adjustment of the digitizer parameters should be done while running in the Acquisition mode **Auto** in one of the application windows (see next section). Any modifications will then immediately be visible in the waveform display.



SC240
SerNo. 20150
Chan # 1
Input Full Scale: 5 V
Offset: -0.00000 V
Coupling: DC 50
Bandwidth Limit: None
Timebase
Sampling Frequency: 1000.000 MS
Max. Memory: 10K
Delay: 0 s
Chan Combination: [✓ 1x] 2x
1x: All channels
Internal Clock:
10000 Samples at
1 ns per Sample

The **Offset** can be modified either by selecting the display of the offset field and typing the desired value followed by a <CR>(Enter), or by clicking on the various fields in the scroll bar. It is also possible to drag the scrollbar.

The **Max Memory** field limits the amount of memory being used to acquire a waveform during the setup phase. 5K or 10K are reasonable to obtain a fast display update rate.

The **Chan Combination** field is of interest for the SC240 only. When **1x** is selected, both inputs are converted independently as 2 separate channels. Clicking on **2x** switches the instrument to the mode where both ADCs are interleaved to a single channel, for sampling rates up to 2 GS/s. The drop-down box below gives a choice of whether the signal at input 1 or 2 is converted.

The **Delay** can be controlled in the same way as the **Offset**. However, it will be ignored when running in the continuous acquisition mode.

The field at the right bottom corner of the Timebase section indicates the settings of the timebase parameters: clock source (internal or external), the number of samples, and the sampling interval. The number of samples is only of interest for the **Single** and **Auto** acquisition modes; it will be ignored when running in the **Continuous** mode.

The pushbutton **Calibrate** requests a recalibration of the current instrument. The buttons **Trigger…** and **Ext Clk…** open additional dialog boxes for the setup of the trigger conditions and the selection of an External Clock option.

5.4 Application Windows & Acquisition Mode

The Base Test application will automatically be opened on the first instrument and **Base Test** will be displayed in the Instrument Control Window. If additional instruments are present, they can be activated by selecting another instrument in the left-hand drop-down list of the Instrument Control Window and selecting the appropriate "Use for/as" value.

You may switch from one application to another at any time, with the "Use for/as" drop-down list. There is a time lag of a couple of seconds because a new firmware file is automatically loaded into the FPGA.

The **Acquisition** control field is present both in the ***Base Test*** (if the Test Category is set to *Data Transfers*), the ***Streamer***, and other optional applications. Thus, it is described in this common section.

The acquisition mode of the analyzer in its digitizer mode is selected using one of four acquisition mode buttons in the **Acquisition** section of the control panel. Available acquisition modes are Stop, Single, Auto and Continuous. The use of each of the acquisition modes is described below. The Single and Auto modes require a valid trigger, i.e. a trigger signal meeting the trigger conditions at a time when the digitizer is armed and ready to acquire data.

- **Stop** will stop the acquisition and hold the latest complete acquisition on the display.
- **Single** mode is used in order to capture one event at the first valid trigger. It freezes the acquisition in the digitizer's memory, and on the display, until the user requests another acquisition. After an acquisition is taken in Single mode, the digitizer will ignore subsequent trigger events until the Single button is pressed again or another acquisition mode is selected. Pressing the single button re-arms the trigger and captures a new acquisition.
- **Auto** mode will acquire and display waveforms according to the trigger settings if a valid trigger is present within a timeout interval. If a valid trigger is not available within this interval, the digitizer generates its own trigger in order to digitize and display whatever signal is at the input at that time.

  Auto mode is typically used to aid in setup when the input signal must be quickly characterized in order to determine proper vertical and trigger settings for Continuous or Single mode acquisitions.
- **Continuous** mode is used to continuously transfer converted data to the on-board DPU (FPGA). In this mode, some digitizer settings such as trigger or delay are ignored. The operation continues until another mode is explicitly selected.

  This mode must be chosen for any operations that are implemented in the FPGA.

5.5 Base Test Application

The Base Test application offers a number of tests for the on-board FPGA of an Analyzer or Streamer Analyzer instrument.

There are 2 test categories available:

- **Registers** tests the connection between the host computer and the FPGA, as well as the connection between the FPGA and the front panel LEDs
- **Data Transfers** tests the connection between the ADC(s) and the FPGA

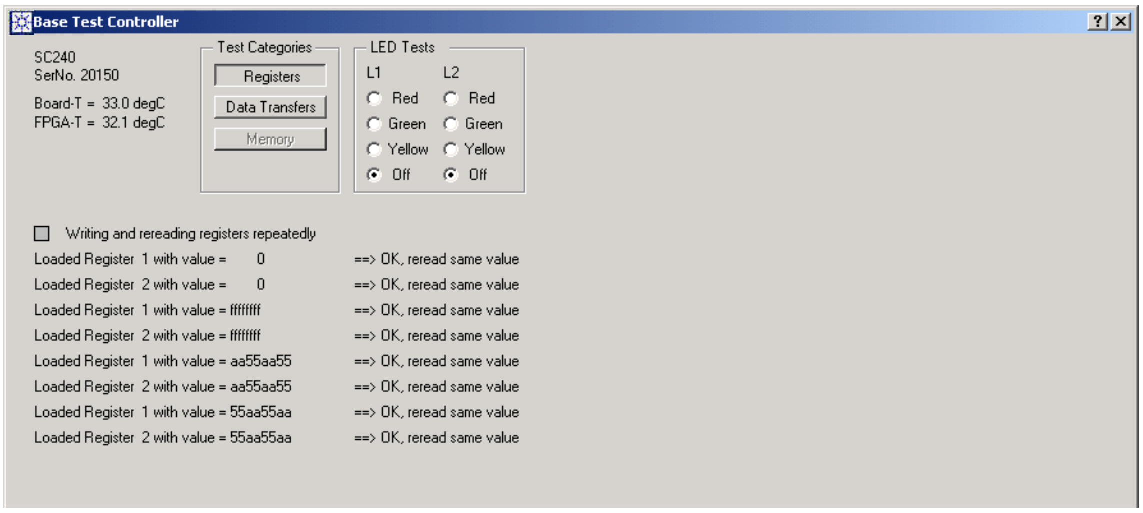
5.5.1 Register Tests

This test writes a number of different data values to a few registers, reads them back, and compares them with the originally written values. The result of the comparison is reported on the screen.

The test is automatically repeated once a second, indicated by a 'blinking' square next to the text 'Writing and rereading registers repeatedly'.

Since the User registers 1 and 2 are present in all AC2x0.bit and SC2x0.bit default firmware files, this test can be executed for all (streaming) analyzer instruments.

Successful tests show "= => OK, reread same value" while a failed tests displays "= => Error, reread xx".



SC240
SerNo. 20150

Board-T = 33.0 degC
FPGA-T = 32.1 degC

Writing and rereading registers repeatedly

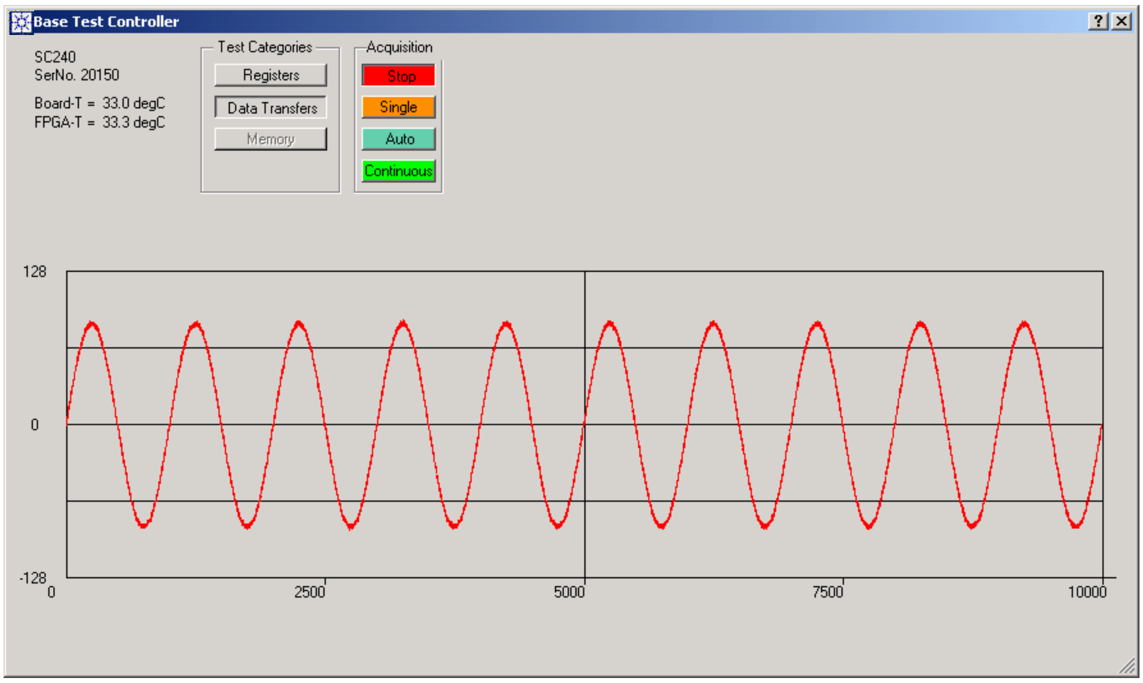
```
Loaded Register  1 with value =        0     ==> OK, reread same value
Loaded Register  2 with value =        0     ==> OK, reread same value
Loaded Register  1 with value = ffffffff     ==> OK, reread same value
Loaded Register  2 with value = ffffffff     ==> OK, reread same value
Loaded Register  1 with value = aa55aa55     ==> OK, reread same value
Loaded Register  2 with value = aa55aa55     ==> OK, reread same value
Loaded Register  1 with value = 55aa55aa     ==> OK, reread same value
Loaded Register  2 with value = 55aa55aa     ==> OK, reread same value
```

In addition, the control of the front panel LEDs by the FPGA can be tested by pushing the radio buttons in the ***LED Tests*** section. The LEDs L1 and L2 should show the requested color.

5.5.2 Data Transfer Tests

This test verifies that ADC data arrive correctly to the FPGA data entry buffer and data entry monitor buffer.

First use the **Auto** mode to set the digitizer to an appropriate state for acquiring your input signal.



SC240
SerNo. 20150

Board-T = 33.0 degC
FPGA-T = 33.3 degC

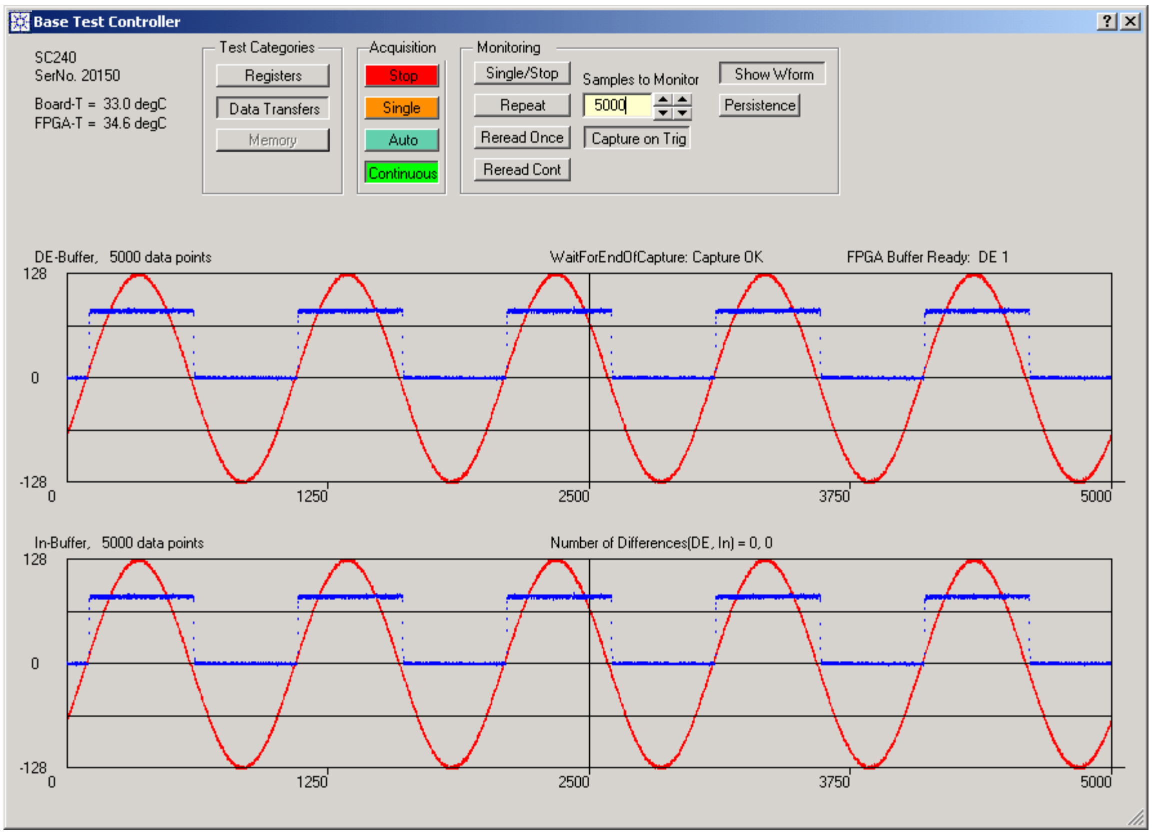
The **Auto** mode does not test the connection to the FPGA, since its waveforms are acquired in the digitizer memory, as opposed to through the FPGA.

For the real test, switch to the **Continuous** mode. A **Monitoring** section appears. The digitizer transfers data continuously to the FPGA, independently of the choices in this section. Select a number of samples to monitor, typically between 1000 and 10000 samples.

- The **Single/Stop** pushbutton is used to capture a single set of waveforms from the FPGA. It then displays the captured waveforms and freezes the display, until it is pushed again for another capture.
- The **Repeat** pushbutton repeats the capture/display sequence indefinitely until the **Single/Stop** pushbutton is depressed.
- The **Reread Once** and **Reread Cont** buttons are for advanced failure diagnostics. They should be ignored unless directed otherwise by the Acqiris Support team.
- The **Capture on Trig** pushbutton directs the monitoring process to capture data only upon the occurrence of a trigger signal at the FPGA input. This mode works when a valid trigger on a signal at the Channel input or the External Trigger input has been set up.

The figure below shows both waveforms of an SC240. The red curve is Channel 1 and the blue one is Channel 2. The upper display represents the waveforms as captured in the Data Entry (DE) section of the FPGA, while the lower one monitors the data stream at the output of the Data Entry section. A text line above the lower waveform display indicates the number(s) of differences between the DE-buffer and the DE-Monitor buffer for each channel. They should always be zero. A non-zero value indicates that the data have not been successfully transferred.

When displaying the waveforms graphically (i.e. with **Show Wform** depressed), the user has the option of selecting **Persistence**. This mode permits the accumulation of many waveforms on the display in the search for intermittent transfer problems. When using persistence display, you must use the **Capture on Trig** in order to stabilize the horizontal position of the waveform.


Base Test Controller

SC240
SerNo. 20150

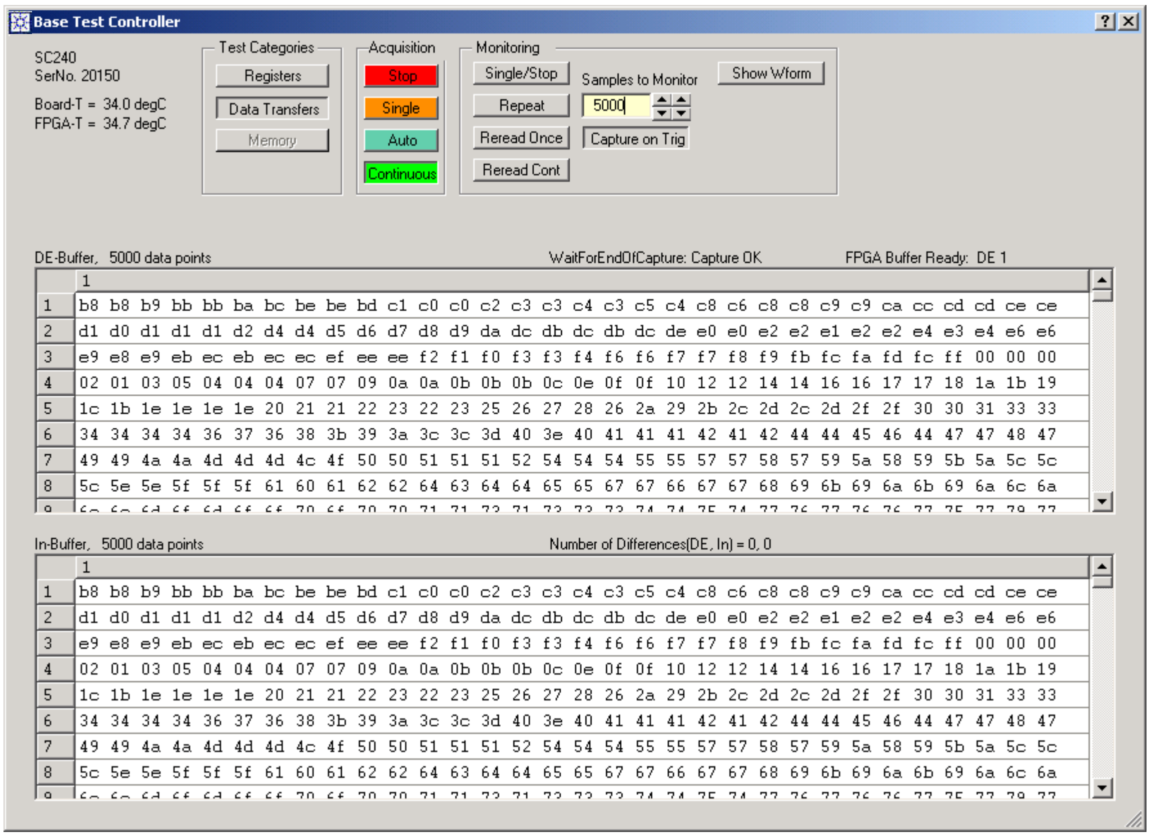
Board-T = 33.0 degC
FPGA-T = 34.6 degC

Samples to Monitor: 5000

DE-Buffer, 5000 data points
WaitForEndOfCapture: Capture OK
FPGA Buffer Ready: DE 1

In-Buffer, 5000 data points
Number of Differences(DE, In) = 0, 0

For a detailed analysis of an observed data transfer problem, the **Show Wform** button can be released as shown in the following figure, to display the data in a numeric, hexadecimal format. When 2 channels are available, the numeric data of channel 1 are displayed first, followed by those of Channel 2.



SC240
SerNo. 20150

Board-T = 34.0 degC
FPGA-T = 34.7 degC

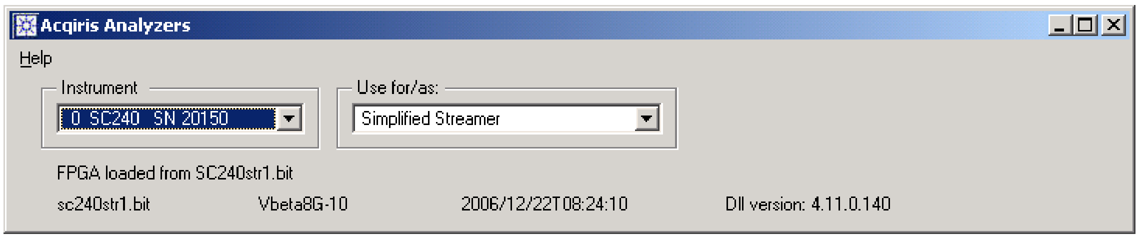
Samples to Monitor: 5000

```
DE-Buffer,  5000 data points            WaitForEndOfCapture: Capture OK        FPGA Buffer Ready:  DE 1
   1
1  b8 b8 b9 bb bb ba bc be be bd c1 c0 c0 c2 c3 c3 c4 c3 c5 c4 c8 c6 c8 c8 c9 c9 ca cc cd cd ce ce
2  d1 d0 d1 d1 d1 d2 d4 d4 d5 d6 d7 d8 d9 da dc db dc db dc de e0 e0 e2 e2 e1 e2 e2 e4 e3 e4 e6 e6
3  e9 e8 e9 eb ec eb ec ec ef ee ee f2 f1 f0 f3 f3 f4 f6 f6 f7 f7 f8 f9 fb fc fa fd fc ff 00 00 00
4  02 01 03 05 04 04 04 07 07 09 0a 0a 0b 0b 0b 0c 0e 0f 0f 10 12 12 14 14 16 16 17 17 18 1a 1b 19
5  1c 1b 1e 1e 1e 1e 20 21 21 22 23 22 23 25 26 27 28 26 2a 29 2b 2c 2d 2c 2d 2f 2f 30 30 31 33 33
6  34 34 34 34 36 37 36 38 3b 39 3a 3c 3c 3d 40 3e 40 41 41 41 42 41 42 44 44 45 46 44 47 47 48 47
7  49 49 4a 4a 4d 4d 4d 4c 4f 50 50 51 51 51 52 54 54 54 55 55 57 57 58 57 59 5a 58 59 5b 5a 5c 5c
8  5c 5e 5e 5f 5f 5f 61 60 61 62 62 64 63 64 64 65 65 67 67 66 67 67 68 69 6b 69 6a 6b 69 6a 6c 6a

In-Buffer,  5000 data points            Number of Differences(DE, In) = 0, 0
   1
1  b8 b8 b9 bb bb ba bc be be bd c1 c0 c0 c2 c3 c3 c4 c3 c5 c4 c8 c6 c8 c8 c9 c9 ca cc cd cd ce ce
2  d1 d0 d1 d1 d1 d2 d4 d4 d5 d6 d7 d8 d9 da dc db dc db dc de e0 e0 e2 e2 e1 e2 e2 e4 e3 e4 e6 e6
3  e9 e8 e9 eb ec eb ec ec ef ee ee f2 f1 f0 f3 f3 f4 f6 f6 f7 f7 f8 f9 fb fc fa fd fc ff 00 00 00
4  02 01 03 05 04 04 04 07 07 09 0a 0a 0b 0b 0b 0c 0e 0f 0f 10 12 12 14 14 16 16 17 17 18 1a 1b 19
5  1c 1b 1e 1e 1e 1e 20 21 21 22 23 22 23 25 26 27 28 26 2a 29 2b 2c 2d 2c 2d 2f 2f 30 30 31 33 33
6  34 34 34 34 36 37 36 38 3b 39 3a 3c 3c 3d 40 3e 40 41 41 41 42 41 42 44 44 45 46 44 47 47 48 47
7  49 49 4a 4a 4d 4d 4d 4c 4f 50 50 51 51 51 52 54 54 54 55 55 57 57 58 57 59 5a 58 59 5b 5a 5c 5c
8  5c 5e 5e 5f 5f 5f 61 60 61 62 62 64 63 64 64 65 65 67 67 66 67 67 68 69 6b 69 6a 6b 69 6a 6c 6a
```

5.6 Base Streamer Application

The behavior of the Base Streamer Firmware can be verified with the program AcqirisAnalyzer.



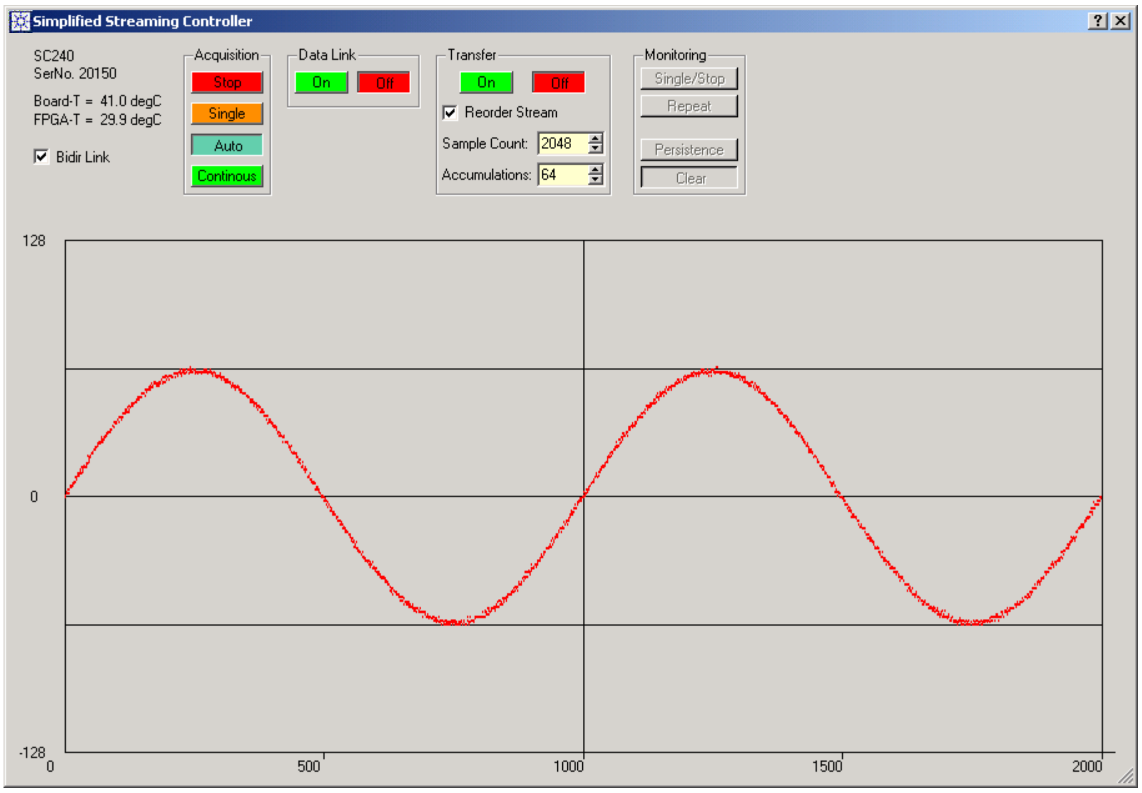
Instrument: 0 SC240 SN 20150

Use for/as: Simplified Streamer

FPGA loaded from SC240str1.bit

sc240str1.bit   Vbeta8G-10   2006/12/22T08:24:10   Dll version: 4.11.0.140

When selecting the Simplified Streamer application, the following window opens:



While in **Auto** Acquisition mode, configure the sample rate and the trigger with the Digitizer Control window:

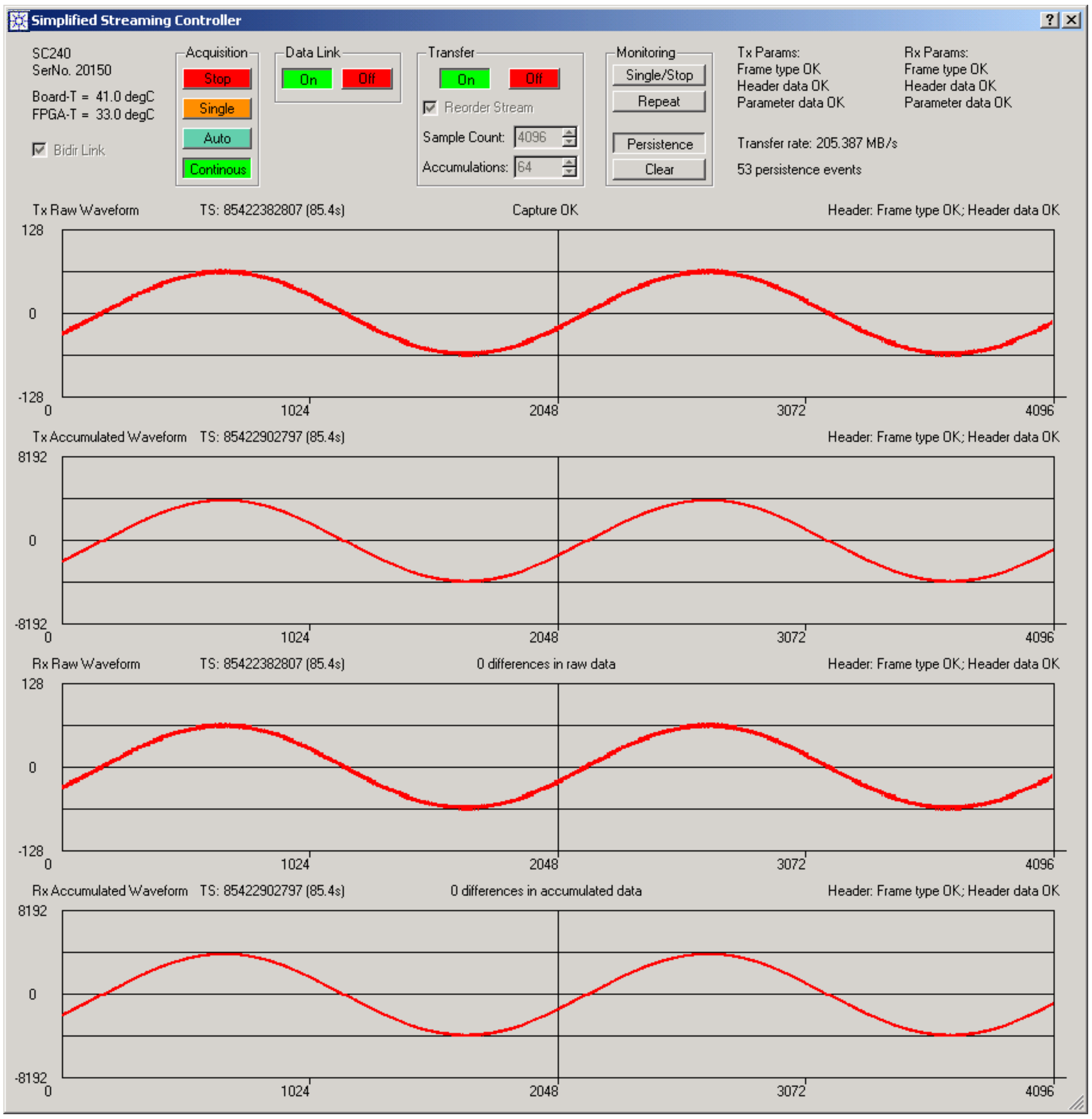
1. The sample rate should be set to 500MS/s or 1GS/s, if **Chan Combination** is set to 1x (non-interleaving of the channels). To interleave the two channels of the SC240, set **Chan Combination** to 2x and select a sample rate of 1GS/s or 2GS/s.
2. Configure the trigger and the vertical range to obtain a valid trigger.

For data streaming proceed as follows:

1. The **Bidir Link** option allows comparison of the sent data with received data. If selected, the Tx signal of the ODL-A must be externally connected to the Rx signal of the ODL-A.
2. Set the **Sample Count** and number of **Accumulations**. Select the choice of **Reorder Stream** if desired and a trigger is being used.
3. Select **Continuous**. Set **Data Link** to **On**. Set **Transfer** to **On**.
4. Set the **Monitoring** mode to either **Repeat** or **Single**.

In the Base Streaming Controller window, the following information is displayed:

1. Upper right portion: Tx status, Rx status, and effective transfer rate.
2. Above the Tx Raw data waveform: status of the data capture. It displays OK if the operation works successfully. Otherwise it displays “TimeOut on Capture”. This could be the case if there is no valid trigger. The timeout can also occur when **Bidir** is selected, but the Tx output of ODL-A is not connected to the Rx input of ODL-A.
3. Above each waveform: The Time Stamp value and the verification status.
4. Above the Rx waveforms, the result of the comparison with the corresponding Tx waveform.


Simplified Streaming Controller

SC240
SerNo. 20150
Board-T = 41.0 degC
FPGA-T = 33.0 degC
Bidir Link

Tx Params:
Frame type OK
Header data OK
Parameter data OK

Rx Params:
Frame type OK
Header data OK
Parameter data OK

Transfer rate: 205.387 MB/s
53 persistence events

Sample Count: 4096
Accumulations: 64
Reorder Stream

Tx Raw Waveform TS: 85422382807 (85.4s) Capture OK Header: Frame type OK; Header data OK

Tx Accumulated Waveform TS: 85422902797 (85.4s) Header: Frame type OK; Header data OK

Rx Raw Waveform TS: 85422382807 (85.4s) 0 differences in raw data Header: Frame type OK; Header data OK

Rx Accumulated Waveform TS: 85422902797 (85.4s) 0 differences in accumulated data Header: Frame type OK; Header data OK

There are restrictions to the operation:

1. The **Sample Count** and the number of **Accumulations** can only be modified when the **Transfer** is turned **Off**.
2. The **Sample Count** is the total number of samples that will be transmitted from channel 1 ***and*** channel 2. To prevent overfilling of the monitoring buffers, the sample count should be limited to 20K. (20 KB of raw data plus 40 KB of accumulated data plus 4 KB of parameter data fills the buffer entirely). The 64 KB are shared for monitoring the three types of frames.
3. The persistence display is available only for the raw data display.

5.7 Streamer Application

The Streamer application permits continuous streaming of converted digitizer data over one or several optical data links to disk storage or computing systems.

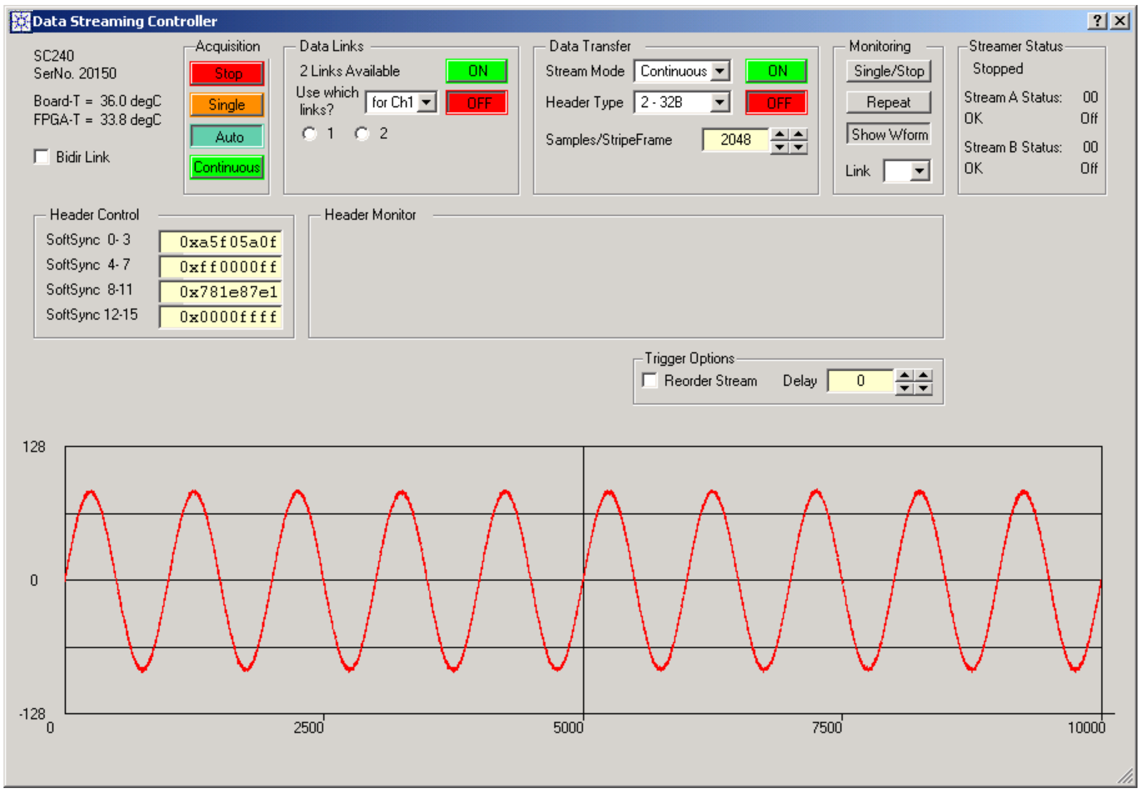
The control panel for this application consists of a control section (upper quarter of window), a frame header control and display section (second quarter of window), and a waveform display section in the lower half of the window.

5.7.1 Digitizer Setup

The control section should be operated left to right. First, you should use the **Acquisition** mode **Auto** (see figure below) to set the digitizer to the desired state, using the Digitizer Control window, as described in the section 5.3, Digitizer Control Window. In particular, you should set the appropriate input full scale range and offset, and the sampling rate. Use **Ext Clk…** to set to an external clock source, if required. The display section shows the acquired waveform interactively.

In this mode, the **Data Links** and **Data Transfer** control sections may be modified, but have no influence on the data acquisition. However, the 2 **ON** buttons should not be depressed yet.

For streaming operations, set the **Acquisition** mode to **Continuous**. The additional required operations are described in the following sections. The status LED at the bottom of the module should turn green and stay permanently green.



Data Streaming Controller

SC240
SerNo. 20150
Board-T = 36.0 degC
FPGA-T = 33.8 degC
Bidir Link

Acquisition: Stop, Single, Auto, Continuous

Data Links: 2 Links Available, ON, OFF
Use which links? for Ch1
1, 2

Data Transfer:
Stream Mode: Continuous
Header Type: 2 - 32B
Samples/StripeFrame: 2048
ON, OFF

Monitoring: Single/Stop, Repeat, Show Wform, Link

Streamer Status: Stopped
Stream A Status: 00 OK Off
Stream B Status: 00 OK Off

Header Control:
SoftSync 0-3: 0xa5f05a0f
SoftSync 4-7: 0xff0000ff
SoftSync 8-11: 0x781e87e1
SoftSync 12-15: 0x0000ffff

Header Monitor

Trigger Options: Reorder Stream, Delay 0

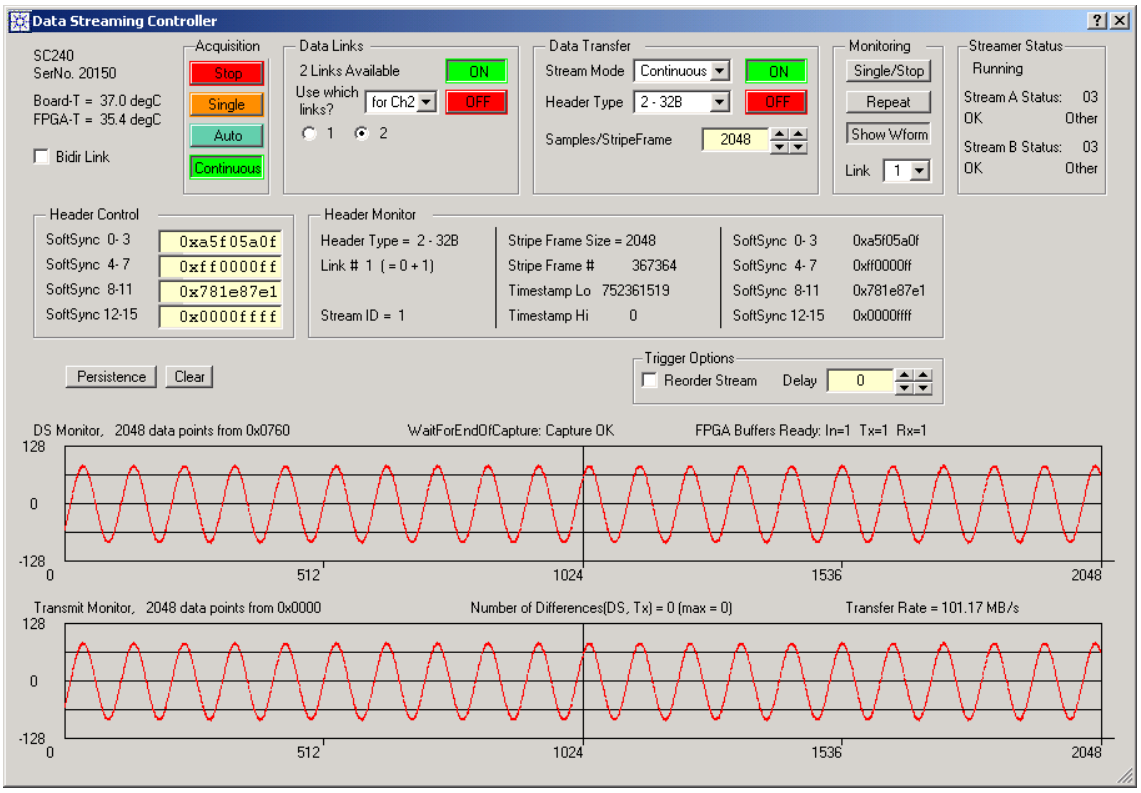
5.7.2 Data Links

The number of available data links and the automatically loaded FPGA firmware depend on the installed optical transceivers. The number of available links will be either 2 (with the dual LC fiber link) or 12 (with the Hi-Rate optical data link option). The Data Links control section displays as many radio buttons as there are available links.

If a single input channel is used, then all links are available to a single data stream. You may choose as many links as available, but at least 1 must be chosen.

If an SC240 is operated in dual channel mode, then the links must be shared between the 2 data streams. The indication **(for Ch1)** then turns into a drop-down box that permits selecting the channel. For each channel, you may select the appropriate data links. If you select a link that was already selected for the other channel, it gets assigned to the currently indicated channel and gets de-assigned from the other channel. Thus, the data link selection must be done carefully, by toggling between the 2 channels.

The transfer of a data stream will occur in a 'round-robin' fashion over all data links assigned to the stream, in increasing order, starting with the lowest data link number.


Data Streaming Controller

SC240
SerNo. 20150
Board-T = 37.0 degC
FPGA-T = 35.4 degC
Bidir Link

Acquisition: Stop, Single, Auto, Continuous

Data Links: 2 Links Available; Use which links? for Ch2; ON; OFF; 1 [✓ 2]

Data Transfer: Stream Mode Continuous; Header Type 2 - 32B; Samples/StripeFrame 2048; ON; OFF

Monitoring: Single/Stop, Repeat, Show Wform; Link 1

Streamer Status: Running
Stream A Status: 03 OK Other
Stream B Status: 03 OK Other

Header Control:
SoftSync 0-3 0xa5f05a0f
SoftSync 4-7 0xff0000ff
SoftSync 8-11 0x781e87e1
SoftSync 12-15 0x0000ffff

Header Monitor:
Header Type = 2 - 32B
Link # 1 (= 0 + 1)
Stream ID = 1
Stripe Frame Size = 2048
Stripe Frame # 367364
Timestamp Lo 752361519
Timestamp Hi 0
SoftSync 0-3 0xa5f05a0f
SoftSync 4-7 0xff0000ff
SoftSync 8-11 0x781e87e1
SoftSync 12-15 0x0000ffff

Persistence, Clear

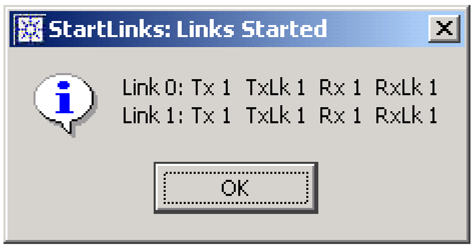
Trigger Options: Reorder Stream; Delay 0

DS Monitor, 2048 data points from 0x0760 WaitForEndOfCapture: Capture OK FPGA Buffers Ready: In=1 Tx=1 Rx=1

Transmit Monitor, 2048 data points from 0x0000 Number of Differences(DS, Tx) = 0 (max = 0) Transfer Rate = 101.17 MB/s

After the data links have been assigned, depress the push button **ON** in the Data Links section. All selected data links are turned on. The program waits until all data links report *ready* and then pops up a small window that reports the link status.

If the links started correctly, the popup window is as shown on the right. There is a status line for each used link; although the numbering of the links in the control section is from 1 to n, the status window counts from 0 to n-1. Tx and TxLk are the 2 bits for the transmit transceiver. They must be 1, indicating that the link is ready in the transmit direction. Rx and RxLk are 2 bits for the receive transceiver. They are currently not used by the Data Streaming application and can be ignored. The program does not check them.


StartLinks: Links Started

Link 0: Tx 1 TxLk 1 Rx 1 RxLk 1
Link 1: Tx 1 TxLk 1 Rx 1 RxLk 1

OK

If the data links did not start correctly, the pop-up window indicates that there was a timeout. Check the Tx and TxLk bits to find out where the problem occurred. If a problem occurs, it might be useful to restart the application, and to adhere to the sequence of operations described in this manual. If the problem persists, check if the correct FPGA file was successfully loaded and that the **Acquisition** is **Continuous**.

Note that the Tx and TxLk bits must work correctly even if there is no physical fiber connected to the front panel connectors (if Rx and RxLk were used, however, a physical fiber connection would be necessary). If the problem still persists, contact Acqiris.

Press **OK** to acknowledge the pop-up window and make it go away.

Before the data links are started, the 2 streamer status LEDs L1 and L2 near the optical connectors should be red. When the data links are started, both LEDs should turn green as soon as all used data links have successfully started. After the data transfer operation has been started (see section below), these LEDs will either be turned off or become yellow.

5.7.3 Data Transfer

The Data Transfer Control section controls the parameters of the data frames.

The **Stream Mode** drop-down list permits the choice of the streaming mode. You have the choice of 'StrtOnTrig', 'Continuous', and 'Triggered'. The three modes are described in section 4.3.1 ***DATA STREAMING MODE***.

The **Header Type** drop-down list permits the choice of the header. You have the choice of '0 – None', '1 – 16B', and '2 – 32B'. The first choice uses no header at all. Only the raw data are embedded into the serial FPDP protocol and transmitted. The second choice adds a 16-byte standard frame header at the beginning of each stripe frame, while the third choice adds the 16-byte standard frame header **and** a 16-byte 'SoftSync' data pattern to the beginning of the stripe frame. For a description of the header structure, please refer to section **4.3.5**, ***STRIPE FRAME STRUCTURE***.

If you choose header type 2, you can modify the 'SoftSync' pattern that is shown in the **Header Control** section, by typing your own values into the fields followed by a <CR>(Enter).

The **Samples/StripeFrame** field can be modified by typing the value followed by a <CR>(Enter) into the field or by using the up/down arrows to the right of the field. The left-hand arrows modify the value by ±16 while the right-hand arrows modify by ±512. This value, which must be a multiple of 16, corresponds to the number of (8-bit) data samples that will constitute the stripe frame, **without** the header. The value of 12800, as shown in the figure of section 5.7.2 ***DATA LINKS***, together with the header 2, will result in a total stripe frame size of 12800+ 32 = 12832 bytes.

The largest accepted total stripe frame size is limited to 16 KB, which will limit the number of samples per stripe frame to 16352, if the 32-byte header is used.

The actual data transfer is started when pushing the **ON** button in the Data Transfer section. At this time, the (green) LEDs L1 and L2 should change color. If a single internal data stream is used (SC210), L1 should turn yellow and L2 should turn off. If both internal data streams are used (SC240), both L1 and L2 should turn yellow.

At this point, the data streaming operation is started and will continue operating until an explicit stop command is received, e.g. by pressing the **OFF** button in the Data Transfer Control section or by aborting the entire application by pressing **Base Test** in the Instrument Control section.

No further interactions between the host computer and the instrument are needed. However, it is possible to observe the data streaming operation without interfering with it, nor degrading its performance. The next section shows how.

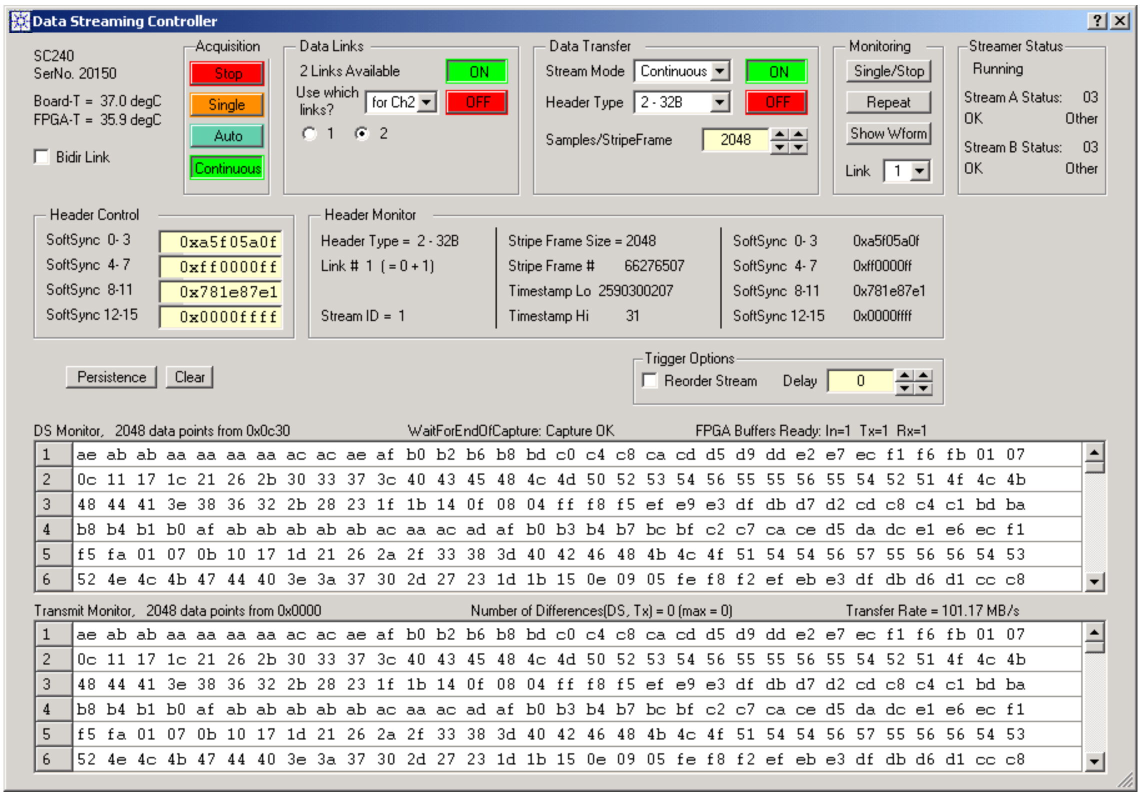
5.7.4 Monitoring

By pressing the button **Single/Stop** in the Monitoring Control section, you obtain a snapshot of a single stripe frame. A copy of the input data, as well as a copy of the transmitted stripe frame, is captured.

The monitoring always refers to a single data link, as chosen in the **Link** drop-down list. Thus, only transmit frames for the chosen data link, together with the associated input data, are captured and shown.

The **Repeat** button simply makes the application repeat the capture/display process, several times per second, until it is explicitly stopped by pressing **Single/Stop**.

The button **Show Wform**, depressed by default, controls if the monitoring buffers are displayed as waveforms (see the previous figure) or as data lists (see below).



SC240
SerNo. 20150
Board-T = 37.0 degC
FPGA-T = 35.9 degC
Bidir Link

2 Links Available
Use which links? for Ch2
1 [✓ 2]

Stream Mode: Continuous
Header Type: 2 - 32B
Samples/StripeFrame: 2048

Link: 1

Running
Stream A Status: 03 OK Other
Stream B Status: 03 OK Other

Header Control	
SoftSync 0-3	0xa5f05a0f
SoftSync 4-7	0xff0000ff
SoftSync 8-11	0x781e87e1
SoftSync 12-15	0x0000ffff

Header Monitor

Header Type = 2 - 32B
Link # 1 (= 0 + 1)
Stream ID = 1

Stripe Frame Size = 2048
Stripe Frame # 66276507
Timestamp Lo 2590300207
Timestamp Hi 31

SoftSync 0-3	0xa5f05a0f
SoftSync 4-7	0xff0000ff
SoftSync 8-11	0x781e87e1
SoftSync 12-15	0x0000ffff

Reorder Stream
Delay 0

DS Monitor, 2048 data points from 0x0c30 WaitForEndOfCapture: Capture OK FPGA Buffers Ready: In=1 Tx=1 Rx=1

```
1  ae ab ab aa aa aa aa ac ac ae af b0 b2 b6 b8 bd c0 c4 c8 ca cd d5 d9 dd e2 e7 ec f1 f6 fb 01 07
2  0c 11 17 1c 21 26 2b 30 33 37 3c 40 43 45 48 4c 4d 50 52 53 54 56 55 55 56 55 54 52 51 4f 4c 4b
3  48 44 41 3e 38 36 32 2b 28 23 1f 1b 14 0f 08 04 ff f8 f5 ef e9 e3 df db d7 d2 cd c8 c4 c1 bd ba
4  b8 b4 b1 b0 af ab ab ab ab ab ac aa ac ad af b0 b3 b4 b7 bc bf c2 c7 ca ce d5 da dc e1 e6 ec f1
5  f5 fa 01 07 0b 10 17 1d 21 26 2a 2f 33 38 3d 40 42 46 48 4b 4c 4f 51 54 54 56 57 55 56 56 54 53
6  52 4e 4c 4b 47 44 40 3e 3a 37 30 2d 27 23 1d 1b 15 0e 09 05 fe f8 f2 ef eb e3 df db d6 d1 cc c8
```

Transmit Monitor, 2048 data points from 0x0000 Number of Differences(DS, Tx) = 0 (max = 0) Transfer Rate = 101.17 MB/s

```
1  ae ab ab aa aa aa aa ac ac ae af b0 b2 b6 b8 bd c0 c4 c8 ca cd d5 d9 dd e2 e7 ec f1 f6 fb 01 07
2  0c 11 17 1c 21 26 2b 30 33 37 3c 40 43 45 48 4c 4d 50 52 53 54 56 55 55 56 55 54 52 51 4f 4c 4b
3  48 44 41 3e 38 36 32 2b 28 23 1f 1b 14 0f 08 04 ff f8 f5 ef e9 e3 df db d7 d2 cd c8 c4 c1 bd ba
4  b8 b4 b1 b0 af ab ab ab ab ab ac aa ac ad af b0 b3 b4 b7 bc bf c2 c7 ca ce d5 da dc e1 e6 ec f1
5  f5 fa 01 07 0b 10 17 1d 21 26 2a 2f 33 38 3d 40 42 46 48 4b 4c 4f 51 54 54 56 57 55 56 56 54 53
6  52 4e 4c 4b 47 44 40 3e 3a 37 30 2d 27 23 1d 1b 15 0e 09 05 fe f8 f2 ef eb e3 df db d6 d1 cc c8
```

Also displayed are the status of the waveform capture, above the upper waveform display frame, the difference between the DS and TX-Monitor buffers (should be zero) and actual effective data transmission rate of the selected data link (above the lower waveform display frame). The transmission rate corresponds to a rate averaged over a millisecond.

If active, the header in the TX Stripe Frame is displayed in the section Header Monitor.

The different fields of the 16-byte standard header are interpreted and displayed. The Header Type, the Link Number, and the Stripe Frame Size (***without*** header) should be identical to the values chosen in the control sections above. The stripe frame number and the time stamps should increase by a significant amount on each capture, depending on the sampling rate, the number of links, and the stripe frame size.

The 'SoftSync' values should correspond to those chosen in the Header Control section.

One of the best ways of monitoring the correct operation of the streaming is to observe the Transfer Rate. A second one is to observe the transmitted waveform; a loss of input signal should be immediately visible.

The data lists (see above) are mostly of use during debugging of the firmware and less for the monitoring of the actual data transfers.

5.7.5 Trigger Options

The trigger controls are available when the stream is configured in mode **Triggered**. It is only available for the Standard(2) streamer version.

Checking the box **Reorder Stream** will reorder the data stream using the high resolution trigger to align the trigger to the first position of the data block in the data stream (the data stream from the ADC is de-multiplexed by a factor 16, thus data blocks of 16 samples are handled in parallel). By checking **Reorder Stream**, the signal monitoring will be precise to a resolution of 1 sample instead of 16 samples when it is unchecked. The **Delay** parameter allows the reorder to start at a desired point after the trigger position. The available valid range is 0 to 128.

5.7.6 Trigger Status

The Trigger Status is available when the stream is configured in mode **Triggered**. The trigger status section displays the trigger status and the trigger timestamp. YES indicates a valid trigger has occurred for the last capture to the monitor buffers.

The timestamp area displays the raw value of the timestamp and the value converted in time in units of seconds.

5.7.7 Streamer Status

The Streamer Status section displays the status of the streamer A and B. It indicates the state of the operation and the status-error code. See section 6.4.8.6 ***STREAMER A/B STATUS REGISTERS***.

5.7.8 Stopping / Modifying the Streamer Operation

In principle, the operations should be undone in the opposite order as they were started, i.e. first Data Transfer **OFF**, then Data Links **OFF**, and finally Acquisition to **Auto**, **Single,** or **Stop**.

If you want to change the header type or the stripe frame size, it should be sufficient to turn only the Data Transfer **OFF**, modify the values, and turn the Transfer **ON** again.

If you want to change the data links, you turn **OFF** the Data Transfer and the Data Links, modify the links, and turn the Data Links and the Transfer **ON** again, in this order.

If you want to change the acquisition parameters, e.g. input range or offset, you need to go back all the way to **Auto**.

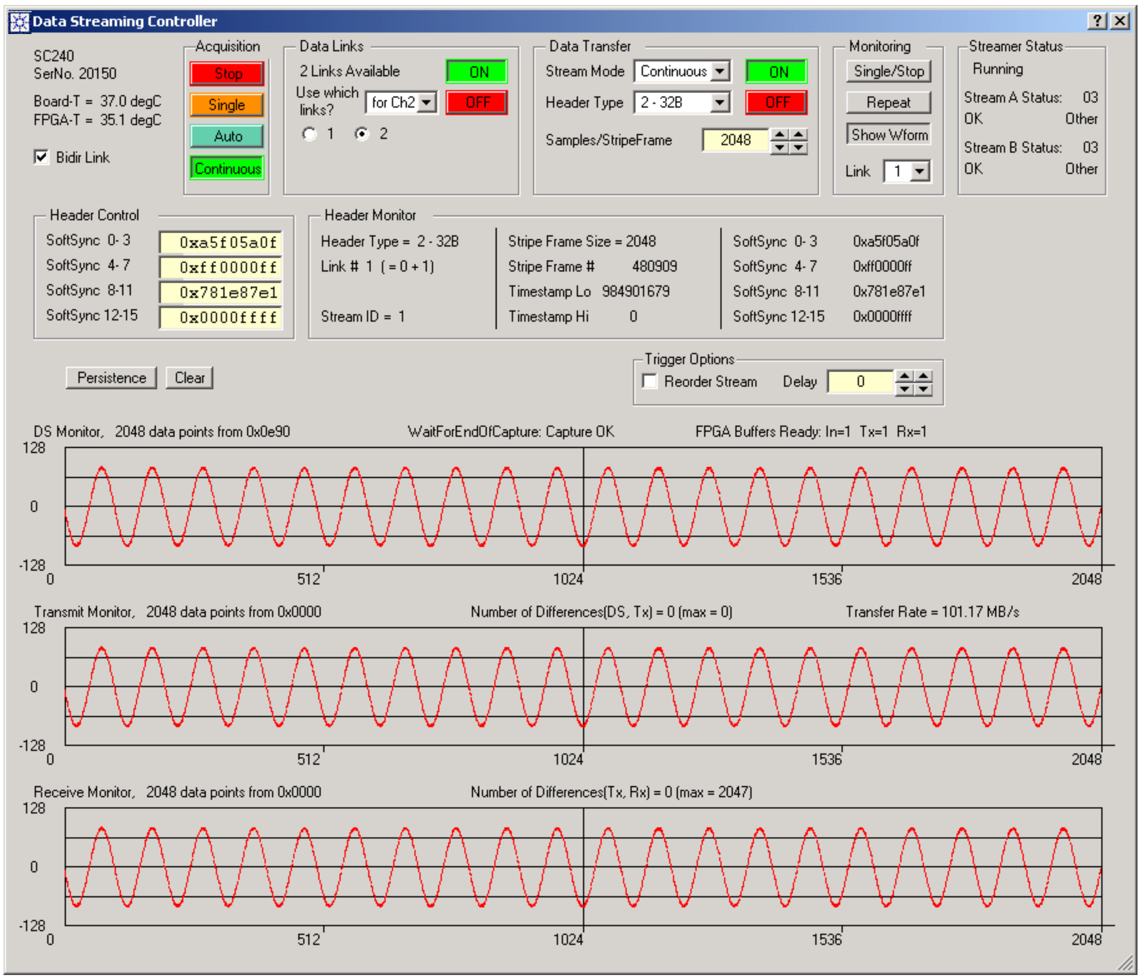
A more violent, but finally even safer way, is to push the **Base Test** button in the Instrument Control and then push the **Streamer** button again. This will bring the streaming operation to a 'screeching' halt and reload the FPGA contents. Of course, you need to go through the setup operations again (other than the digitizer configuration, which is remembered). This method ensures that you always start in exactly the same way, while the other, shorter procedures might suffer from side effects when turning off the transfer or the links (not supposed to happen, but still not impossible).

5.7.9 Verification of the Transmission

For a rapid and simple verification of the optical link hardware and of the transmission, the streamer firmware implements the capability of monitoring data capture on the Receiver side of the link. Connecting the Tx output to the Rx input permits verifying that received data are consistent with the transmitted data.

In the case of the dual link streamer, the Tx output of the ODLA must be connected to the Rx input of the ODLB and the Tx output of the ODLB to the Rx input of the ODLA. This is simply done by using a commonly available dual fiber cable.

This special mode is enabled by checking the box **Bidir Link** located at the upper left side of the Data Streaming controller window.



Data Streaming Controller

SC240
SerNo. 20150
Board-T = 37.0 degC
FPGA-T = 35.1 degC
☑ Bidir Link

Acquisition: Stop, Single, Auto, Continuous

Data Links
2 Links Available — ON / OFF
Use which links? for Ch2 — ○ 1 ● 2

Data Transfer
Stream Mode: Continuous — ON / OFF
Header Type: 2 - 32B
Samples/StripeFrame: 2048

Monitoring: Single/Stop, Repeat, Show Wform
Link: 1

Streamer Status
Running
Stream A Status: 03 OK Other
Stream B Status: 03 OK Other

Header Control

SoftSync 0-3	0xa5f05a0f
SoftSync 4-7	0xff0000ff
SoftSync 8-11	0x781e87e1
SoftSync 12-15	0x0000ffff

Header Monitor
Header Type = 2 - 32B
Link # 1 (= 0 + 1)
Stream ID = 1
Stripe Frame Size = 2048
Stripe Frame # 480909
Timestamp Lo 984901679
Timestamp Hi 0

SoftSync 0-3	0xa5f05a0f
SoftSync 4-7	0xff0000ff
SoftSync 8-11	0x781e87e1
SoftSync 12-15	0x0000ffff

Persistence | Clear

Trigger Options
☐ Reorder Stream Delay 0

DS Monitor, 2048 data points from 0x0e90 WaitForEndOfCapture: Capture OK FPGA Buffers Ready: In=1 Tx=1 Rx=1

Transmit Monitor, 2048 data points from 0x0000 Number of Differences(DS, Tx) = 0 (max = 0) Transfer Rate = 101.17 MB/s

Receive Monitor, 2048 data points from 0x0000 Number of Differences(Tx, Rx) = 0 (max = 2047)

Also displayed are the status of the waveform capture, above the lower waveform display frame, and the difference between the TX- and RX-Monitor buffers (should be zero).

6. Programming the Firmware

Please refer to the **Programmer's Guide** for explanations on which programming environments are supported and how to use them. The **Programmer's Reference Manual** lists all available functions and explains the use of their parameters. The function **Acqrs_logicDeviceIO**, essential for the control of the registers in the data processing unit (DPU), is described there.

The sample code lines below assume a C/C++ environment. They do not check the return value of the **AcqrsXX_…** functions. In real applications, you should always check the return values of functions.

The first part of this chapter describes programming aspects that are common to all applications. The second part contains sections that are applicable to specific firmware applications. They are marked as such.

6.1 Programming Aspects Common to All SC2x0 Applications

6.1.1 Accessing the DPU Registers

All operations in the data processing unit are controlled through registers that are implemented in the FPGA firmware. They are accessed through the function **Acqrs_logicDeviceIO**. One of its arguments is the *registerID* that identifies which register is written to or read from. The AC2x0 and SC2x0 analyzers accept values between 0 and 127, for a total of 128 user-accessible registers. Each application typically uses only a small subset of them. To find which ones, please refer to the appropriate section later in this chapter.

In order to make the code samples in the subsequent section more readable, the following 2 functions are defined:

```
long FPGARead(long regID, long nbrValues, long* dataArrayP)
{
return Acqrs_logicDeviceIO(instrumentID, "Block1Dev1", regID,
                        nbrValues, dataArrayP, 0, 0);
}

long FPGAWrite(long regID, long nbrValues, long* dataArrayP)
{
return Acqrs_logicDeviceIO(instrumentID, "Block1Dev1", regID, nbrValues,
                        dataArrayP, 1, 0);
}
```

6.1.2 Register Definitions

A number of variable and register definitions that may apply to Agilent Acqiris-supplied applications are used in this chapter:

```
// General FPGA-related constants
enum FPGAioConstants
{
       DDR0BufAddress      = 0x00, // Address of dynamic RAM bank 0
       DDR1BufAddress      = 0x01, // Address of dynamic RAM bank 1
       SRAMBufAddress      = 0x04, // Address of static RAM
       DeBufferAddress     = 0x08, // Address of data entry buffer
};

// Register addresses in Acqiris FPGA-firmware
enum FPGAregisters
{
// Registers in region "Acqiris reserved"
// These registers are common to all Acqiris-supplied FPGA designs
       ReadAddrReg         =   0, // Indirect Access Port
       StartAddrReg        =   1, // Start address within block
       BufferIDReg         =   2, // Buffer Identifier Register

       FPGACtrlReg         =   3, // FPGA control register
       CodeProtectReg      =   4, // FPGA code protection register
       FPGAStatusReg       =   6, // FPGA status register
       TemperatureReg      =   7, // Temperature register
       DECtrlReg           =   8, // Data Entry (DE) control register
```

```
        TriggerCtrlReg      =  12, // 1 ns Trigger Manager control register
        TriggerStatusLo     =  13, // 1 ns Trigger Manager status (lo part)
        TriggerStatusHi     =  14, // 1 ns Trigger Manager status (hi part)

        FPIOlinkReg         =  32, // Control for 2 Front Panel PIO lines
        DACctrlReg          =  33, // Control of 16-bit DAC
        LEDReg              =  34, // LED control register

        uDB_IOctrlReg       =  36, // Control register for microDB link
        uDB_IOoutReg        =  37, // Output  register for microDB link
        uDB_IOinReg         =  38, // Input   register for microDB link
};
```

6.1.3 Device Initialization

Before any device can be used, each device must be initialized with a call to the function **Acqrs_InitWithOptions**, typically with *Identification by Order Found*. For details, please refer to section **3.2, *DEVICE INITIALISATION*** in the **Programmer's Guide**.

The function returns the **instrumentID,** (whose value will be different for each device), which must be subsequently used in any other function call to the device.

The sample code below assumes that there is a single instrument attached to the computer. The **Programmer's Guide** shows sample code for more complex situations.

```
ViSession instrumentID;
Acqrs_InitWithOptions("PCI::INSTR0", VI_FALSE, VI_FALSE, "", &instrumentID);
```

This initialization function will automatically recognize an SC240/210 and load the default FPGA configuration file ***SC240.bit*** or ***SC210.bit***. This FPGA configuration supports some device tests, but not data streaming or any other specialized application. A special configuration file must be loaded, as shown in the next section.

6.1.4 Loading an FPGA Configuration File

The data streaming firmware is contained in configuration files with the extension **.bit**. The table at the beginning of chapter 4 ***FIRMWARE*** shows the available possibilities.

The appropriate file must be loaded explicitly with the following function call:

```
Acqrs_configLogicDevice(instrumentID, "Block1Dev1",
          "SC210stream2.bit", 1);
```

The string "Block1Dev1" identifies the FPGA on the board. For all AC2x0 and SC2x0, it is constant. The configuration files should be situated in the same directory as the application or the one indicated by the fpgaPath in the AqDrv4.ini file. The loading process typically takes several seconds.

6.1.5 Sequence of Data Processing Operations

Start: The data processing operations must be configured and started in a well-defined way:

a) Configure the digitizer
b) Start data conversion and start streaming data to DPU
c) Configure the DPU for the requested operation
d) Start the DPU operation
e) Optionally: Interact with the DPU during operation, or monitor a continuous data transfer operation.

Stop: The data processing operations should be stopped in the opposite order:

a) Stop the DPU operation
b) Stop the data conversion and data streaming to the DPU

Each step is described in the subsequent sections in more detail.

6.1.5.1 Digitizer Configuration

The digitizer section must be appropriately configured for the expected input signal, as shown in the sample code below:

```
long fullScale = 1.0;              // full scale range = 1.0 V
double sampInterval = 1.0e-9;      // 1 GS/s sampling rate
double delay = 0.0;                // delay ignored, set it to zero
AcqrsD1_configMode(instrumentID, 1, 0, 0);
AcqrsD1_configVertical(instrumentID, 1, fullScale, 0.0, 3, 0);
AcqrsD1_configHorizontal(instrumentID, sampInterval, delay);
```

Comments:

- The function **AcqrsD1_configMode** must be used to set the instrument to the mode *stream data to DPU* (mode = 1), whereby the instrument will **not** stop upon the receipt of a trigger, but continue data acquisition until explicitly stopped by a software command.
- The function **AcqrsD1_configVertical** configures the signal input channel (1). The full scale is set to 1.0 (V), the offset to 0.0 (V), the input coupling to 50 Ohms (3), and the bandwidth limit to *no limit* (0). These values should be appropriately modified for the actual input signal.
- The function **AcqrsD1_configHorizontal** sets the sampling rate. The value *delay* is ignored, since it is only needed in the *normal* mode, where the trigger stops an acquisition.

For further details, please refer to the section 3.3 ***DEVICE CONFIGURATION*** in the **Programmer's Guide**.

6.1.5.2 Starting Data Conversion

The function **AcqrsD1_configMode** shown in the previous section, with mode = 1, already has configured the SC240/210 to streaming mode. The following code starts the operation of the data converters:

```
AcqrsD1_acquire(instrumentID);
```

Upon receipt of this command, the driver software translates the previously received configuration parameters into the appropriate register values and loads them into the SC240/210. Finally, it transmits a start command, whereupon the SC240/210 starts digitizing the signal at the input channel(s) and transferring the converted data stream to the DPU, until an explicit software command stops it.

The data streaming firmware in the DPU will stay idle until further configuration commands are received, as shown in the next sections.

6.1.5.3 Configuring the Data Processing Unit

The DPU configuration is application specific. Typical configuration steps may be:

- Enabling the internal clocks of the FPGA (DCMs)

```
    long fpgaCtrl = 0;
    WriteFPGA(FPGACtrlReg, 1, &fpgaCtrl);   // First disable everything
    fpgaCtrl |= 0x00ff0000;                 // Enable all DCMs
//  fpgaCtrl |= 0x00000100;  // Enable readout in Big-Endian format
    WriteFPGA(FPGACtrlReg, 1, &fpgaCtrl);
    Sleep(10);                             // Wait some time
```

- Starting the Data Entry (DE) interface of the FPGA

```
    long deCtrl = 0x00000000;
    WriteFPGA(DECtrlReg, 1, &deCtrl);
    deCtrl = 0x80000000;
    WriteFPGA(DECtrlReg, 1, &deCtrl);
```

- Configuring the internal operation of the DPU; code for the FFT Spectrometer firmware can be seen in section 6.3.1.

The order of the function calls may be important.

6.1.5.4 Monitoring

If the firmware runs completely autonomously, e.g. if the result of the calculations is transmitted through the front-panel DAC or the serial links, the host processor need not interact with the digitizer anymore, until the on-going processing is explicitly stopped. It may be useful to monitor the card by occasionally reading some monitoring data from it. This is entirely application-specific. Please refer to the appropriate section later in this chapter.

If the firmware does not run autonomously, it generates data that must periodically be read by the host computer. The data read operation is similar to a monitoring read operation. Again, it is application-specific. Please refer to the appropriate section later in this chapter (6.2.2.4 for the Base Streamer or 6.3.2.4 for the Streamer).

6.1.5.5 Stopping the DPU Operation

The operations required to stop the DPU is application specific. Please refer to the appropriate section later in this chapter (6.2.2.5 for the Base Streamer or 6.3.2.5 for the Streamer).

6.1.5.6 Stopping the Data Conversion

The data acquisition is stopped with the function **AcqrsD1_stopAcquisition**.

```
AcqrsD1_stopAcquisition(instrumentID);
```

6.1.6 Reading the FPGA Temperature

The temperature of the FPGA is of interest, since failures might occur at high temperatures. If the firmware implements temperature monitoring (which is the case for all Agilent Acqiris-supplied firmware), then use the following code to read the temperature whenever required:

```
long tReg;
FPGARead(TemperatureReg, 1, &tReg);
if ((tReg & 0x8000) == 0)
{
     tReg |= 0x8000; // Monitoring is NOT enabled, enable it first
     FPGAWrite(TemperatureReg, 1, &tReg);
     FPGARead( TemperatureReg, 1, &tReg);
}

long tValue = (tReg & 0x1fff);       // signed value
if(tValue > 0x1000)tValue =  tValue - 0x2000;
double temperature = tValue * 0.0625;
```

The temperature value is in centigrade. Temperatures up to 85^{o} C are acceptable. If the temperature exceeds this value, the cooling should be improved (or if possible, the dissipation of the operating firmware reduced by design changes). In extreme cases, contact Acqiris.

6.2 Programming the Base Streamer Application

Note: The following code examples assume that the SLC receiver is not used. For an example including the receiver, see the GetStartedSC240BaseStrmVC example source code, written in C++ and available in the vc folder of your Acqiris installation.

6.2.1 Register Definitions

In addition to the constants defined in section **6.1.2 *REGISTER DEFINITIONS***, a number of constants that are specific to the base streamer application are used in this chapter:

```
// Base Streamer Firmware related constants
enum StreamerIoConstants
{
       TxMonitorAddress    = 0x10, // Address of transmit monitoring buffer
       RxMonitorAddress    = 0x20, // Address of receive  monitoring buffer
};

// Register addresses in Acqiris Streamer firmware
enum FPGARegisters
{
```

```
// Registers in region "User"
// The registers defined below are used in the Base Streamer firmware.
// In user-defined firmware designs, these registers may be assigned
// in an arbitrary way.
      MainCtrlReg         =  64, // Main Control for ALL Acqiris applications
      TxMonStatusReg      =  66, // Status of the transmit monitoring buffer
      RxMonStatusReg      =  67, // Status of the receive  monitoring buffer

      StrmStatusReg       =  68, // Status of the streamer
      StrmConfReg         =  73, // Configuration of the streamer
      SLC0CtrlReg         =  80, // SLC control register for data link 0
      SLC0StatusReg       =  81, // SLC status  register for data link 0
};
```

6.2.2 Streaming Operation

After having configured the FPGA, see 6.1.4 ***LOADING AN FPGA CONFIGURATION FILE*** the following operations can be done:

Start: The streaming operation must be configured and started in a well-defined way:

a) Configure the digitizer; Please refer to section **6.1.5.1 *DIGITIZER CONFIGURATION***

b) Start data conversion and start streaming data to the DPU; Please refer to section **6.1.5.2 *STARTING DATA CONVERSION***

Then continue as described in the next few sections to:

c) Configure the DPU for the requested operation

d) Configure the serial data link and start its operation

e) Configure the data transfer parameters and start actual data transfer

f) Optionally: Interact with the DPU during operation, or monitor a continuous data transfer operation.

Stop: The data processing operations should be stopped in the opposite order:

a) Stop the data transfer

b) Stop the serial data link

c) Stop the data conversion and data streaming to the DPU

6.2.2.1 Configuring the Data Processing Unit

For data streaming, the DPU must be configured with the following commands:

```
    // Initialize the 1 ns trigger manager
    long value = 4; // initialize the DCM
    FPGAWrite(TriggerCtrlReg, 1, &value);
    Sleep(10);      // Wait some time
    value = 8;      // Reset the timestamp counter
    FPGAWrite(TriggerCtrlReg, 1, &value);
    Sleep(10);      // Wait some time
    value = 1;      // Start the 1ns Trigger Manager
    FPGAWrite(TriggerCtrlReg, 1, &value);

    // Turn on the PLL reference clock for the Rocket IO
    AcqrsD1_setAttributeString(currentID, 0, "odlTxBitRate", "2.5G");
    Sleep(10);      // Wait for PLL to stabilize
    // Set the DCM enable bits in the FPGA configuration register
    long fpgaCtrl = 0;
    FPGAWrite(FPGACtrlReg, 1, &fpgaCtrl);     // First disable everything
    fpgaCtrl |= 0x00ff0000;                   // Enable all DCMs
    FPGAWrite(FPGACtrlReg, 1, &fpgaCtrl);
    Sleep(10);                                // Wait some time
```

```
// Start the DE interface
long deCtrl = 0x80000000;
FPGAWrite(DECtrlReg, 1, &deCtrl);


// Wait until the DE clock is ready
bool ready = false;
long timeout = 100;  // With 1 ms 'Sleep --> 100 ms timeout
while ((!ready) && (timeout > 0))
{
     timeout--;
     long fpgaStatus;
     FPGARead(FPGAStatusReg, 1, &fpgaStatus);
     ready = ((fpgaStatus & 0x00100000) != 0);
     if (!ready)
          Sleep(1);
}

if (timeout <= 0)
     ;    // Handle DE-Clock timeout error here!
// Clear the Main Control Register
long mainCtrl = 0;
FPGAWrite(MainCtrlReg, 1, &mainCtrl);
```

6.2.2.2 Configuring/Starting the Data Link

The serial data links are internally numbered from 0 to 1, or 0 to 11, depending on whether the *Standard*(2) or the *HRODL(12)* link option is present. The number of available data links can be retrieved from the instrument driver with the function:

```
long nbrLinks;
AcqrsD1_getInstrumentInfo(instrumentID, "LogDevDataLinks", &nbrLinks);
```

The Base Streamer application only uses the first data link, with an internal number of zero. However, the configuration of this data link is different for the *Standard(2)* and the *HRODL(12)* link options.

The following configuration code is required:

```
if (nbrLinks <= 2)
{
     // For Standard(2) hardware:
     // Tx polarity = default, Rx polarity = inverted
     // Rx FIFO threshold = 0x3f
     // Tx Enable = 1, Rx Enable = 0
     slcCtrl    = 0x023f0001L;
}
else
{
     // For HRODL(12) hardware:
     // Tx polarity = inverted, Rx polarity = inverted
     // Rx FIFO threshold = 0x3f
     // Tx Enable = 1, Rx Enable = 0
     slcCtrl    = 0x033f0001L;
}

FPGAWrite(SLC0CtrlReg, 1, &slcCtrl);
```

The software must wait until the link is actually ready and running, with this code:

```
bool ready = false;
long timeout = 500;  // With 10 ms 'Sleep' --> 5 s timeout

while ((!ready) && (timeout > 0))
{
     timeout--;
     long slcStatus;
     // Is the transmitter ready?
     FPGARead(SLC0StatusReg, 1, &slcStatus);
     ready = (slcStatus & 0x14) == 0x14;
     if (!ready) Sleep(10);
}

if (timeout <= 0)
     ;    // Handle link timeout error here!
```

6.2.2.3 Configuring/Starting Data Transfer

The following code is needed to configure and start the data transfer:

```
// Configure the frame size
long strmConf = NbrSamples / 16 - 1;
FPGAWrite(StrmConfReg, 1, &strmConf);

// Configure and start the Streamer
// Transfer on = 1
// Use bidirectional link = 0
long mainCtrl = 0x00000100L;
// Set the number of accumulations
mainCtrl |= (NbrAccum - 1) << 24;
FPGAWrite(MainCtrlReg, 1, &mainCtrl);
```

Comments:

- The variable **NbrSamples** indicates the number of data samples to include in the **Raw Data** and **Accumulated Data** frames
- The variable **NbrAccum** indicates the number of accumulations to perform for the **Accumulated Data** frame

6.2.2.4 Monitoring the Data Transfer Operation

After the start of the streaming operation, the SC240 runs *without* any further software intervention. To monitor the progress, however, it is possible to sample the contents of two intermediate buffers.

A monitoring cycle consists of making the request for capturing data into one or two monitor buffers, waiting for the capture to be terminated, and reading the corresponding monitor buffers.

The following sample code assumes that only the TX-Monitor buffer is captured and read.

(1) Requesting Capture

Requesting the capture of a monitor buffer involves clearing the corresponding bit in the main control register and setting it to 1 again:

```
// Clear the capture bits in the main control register
long mainCtrl = 0;
FPGARead(MainCtrlReg, 1, &mainCtrl);
mainCtrl &= ~0x6000;
FPGAWrite(MainCtrlReg, 1, &mainCtrl);

// Set the capture bit for the TX-Monitor buffer
mainCtrl |= 0x2000;
FPGAWrite(MainCtrlReg, 1, &mainCtrl);
```

(2) Waiting for End of Capture

When the capture has completed successfully, the MSB of the status register corresponding to the captured buffer goes to 1. For the TX-Monitor buffer, this is User Register 66. For the RX-Monitor buffer, it is User Register 67.

Use the following code to wait for the capture to terminate:

```
bool ready = false;
long timeout = 100;  // With 10 ms 'Sleep' --> 1 s timeout
long status;

while (!ready && timeout > 0)
{
     --timeout;
     FPGARead(TxMonStatusReg, 1, &status);
     bool ready = (status & 0x80000000) != 0;
     if (!ready) Sleep(10);
}

if (timeout <= 0)
     ;    // Handle capture timeout error here!
```

(3) Reading Monitor Buffers

While reading the monitor buffers is as simple as any indirect addressing operation, interpreting the returned data is slightly more complex, due to the heterogenous nature of the monitoring buffers. Each monitor buffer contains three data frames: a **Raw Data** frame, an **Accumulated Data** frame, and a **Parameter Data** frame. Each frame in turn is preceded by a header of 128 bits, containing the frame type, the frame timestamp, and some arbitrary parameter data. The payload data of the **Raw Data** frame consists of **NbrSamples** channel-interleaved 8-bit signed data samples, while the **Accumulated Data** frame contains **NbrSamples** channel-interleaved 16-bit signed values. In both cases, **NbrSamples** corresponds to the number of data samples per frame as configured in section **6.2.2.3** ***CONFIGURING/STARTING DATA TRANSFER***. The Parameter Data frame finally contains 256 blocks of 128 bits of parameter data. See section **4.2** ***BASE STREAMING FIRMWARE*** for a complete description of the format of the various frame types.

The example code in this section only shows how to read the monitored data into a memory buffer. For an example of how to interpret the read-out data, see the GetStartedSC240BaseStrmVC example application.

The following code reads out the TX-Monitor buffer:

```
// Calculate the size of the monitor buffer
long NbrLongs =
       4                   // for 'raw waveform' frame header
     + NbrSamples / 4      // for 'raw waveform' samples
     + 4                   // for 'accumulated waveform' frame header
     + NbrSamples / 2      // for 'accumulated waveform' samples
     + 4                   // for 'parameter data' header
     + 1024;               // for 'parameter data'

long txMonData[NbrLongs];       // memory buffer to hold the data

long startAddr = 0;             // read from the beginning of the buffer
long bufferID = TxMonitorAddress;    // ID of the TX-Monitor buffer

// Set up indirect addressing
FPGAWrite(StartAddrReg, 1, &startAddr);
FPGAWrite(BufferIDReg, 1, &bufferID);

// Read out the monitor data
FPGARead(ReadAddrReg, NbrLongs, txMonData);
```

6.2.2.5 Stopping Data Transfer

The data transfer to the serial data link is stopped by clearing the ‘framing’ bit in the main control register.

```
// Disable framing in the main control register
long mainCtrl;
FPGARead(MainCtrlReg, 1, &mainCtrl);
mainCtrl &= 0xfffffeff;
FPGAWrite(MainCtrlReg, 1, &mainCtrl);
```

6.2.2.6 Stopping the Data Link

The data link is stopped by writing 0 into its control register.

```
long lnkCtrl = 0;
FPGAWrite(SLC0CtrlReg, 1, &lnkCtrl);
```

6.3 Programming the Streamer Application

Programming this application is relatively complex. Before you copy the code samples shown in the subsequent sections, consider using directly the Get StartedSC240StreamerVC example..

6.3.1 Register Definitions

A number of constants that are specific to the streamer application are used in this chapter:

```
// Streamer Firmware related constants
enum StreamerIoConstants
{
       DsMonitorAddress    = 0x0c, // Address of data stream monitoring buffer
       TxMonitorAddress    = 0x10, // Address of transmit monitoring buffer
       RxMonitorAddress    = 0x20, // Address of receive  monitoring buffer
};

// Register addresses in Acqiris Streamer firmware
enum FPGAregisters
{
// Registers in region "User"
// The registers defined below are used in the Data Streamer firmware.
// In user-defined firmware designs, these registers may be assigned
// in an arbitrary way.
```

```
        MainCtrlReg          =  64, // Main Control for ALL Acqiris applications
        DsBufCtrl            =  65, // Control for data stream monitoring buffer
        TxBufCtrl            =  66, // Control for transmit monitoring buffer
        RxBufCtrl            =  67, // Control for receive  monitoring buffer

        StreamerStatusGlob   =  68, // Status of all data links
        StreamerStatusSrcA   =  69,
        StreamerStatusSrcB   =  70,
        StreamerBERT         =  71,

        StreamerConfigGlob   =  72, // Link Configuration
        StreamerConfigSrcA   =  73,
        StreamerConfigSrcB   =  74,

        StreamerSoftSyncW0   =  76, // SoftSync Configuration
        StreamerSoftSyncW1   =  77,
        StreamerSoftSyncW2   =  78,
        StreamerSoftSyncW3   =  79,

        StreamerSLCbase1     =  80, // Base address register(BAR) for data link 1

        SLCbaseOffset        =   4, // SLC: Offset to the BAR of the next link
        SLCctrlOffset        =   0, // SLC control register offset (wrt to base)
        SLCstatusOffset      =   1, // SLC status  register offset (wrt to base)
        SLCsignalOffset      =   2, // SLC signal  register offset (wrt to base)
};
```

6.3.2 Streaming Operation

After having configured the FPGA, see 6.1.4 ***LOADING AN FPGA CONFIGURATION FILE*** the following operations can be done.

Start: The streaming operation must be configured and started in a well-defined way:

a) Configure the digitizer; Please refer to section **6.1.5.1 *DIGITIZER CONFIGURATION***

b) Start data conversion and start streaming data to the DPU; Please refer to section **6.1.5.2 *STARTING DATA CONVERSION***

Then continue as described in the next few sections to:

c) Configure the DPU for the requested operation

d) Configure the serial data links and start their operation

e) Configure the data transfer parameters and start actual data transfers

f) Optionally: Interact with the DPU during operation, or monitor a continuous data transfer operation.

Stop: The data processing operations should be stopped in the opposite order:

a) Stop the data transfers

b) Stop the serial data links

c) Stop the data conversion and data streaming to the DPU

6.3.2.1 Configuring the Data Processing Unit

For data streaming, the DPU must be configured with the following commands:

```
      // Initialize the 1 ns trigger manager
      long value = 4; // initialize the DCM
      WriteFPGA(TriggerCtrlReg, 1, &value);
      Sleep(10);      // Wait some time
      value = 8;      // Reset the timestamp counter
      WriteFPGA(TriggerCtrlReg, 1, &value);
      Sleep(10);      // Wait some time
      value = 1;      // Start the 1ns Trigger Manager
      WriteFPGA(TriggerCtrlReg, 1, &value);
```

```
// Turn on the PLL reference clock for the serial data links
AcqrsD1_setAttributeString(instrumentID, 0, "odlTxBitRate", "2.5G", 0);
Sleep(10);      // Wait 10 ms for PLL to stabilize

// Set the DCM enable bits in the FPGA control register
long fpgaCtrl = 0;
FPGAWrite(FPGACtrlReg, 1, &fpgaCtrl);     // Clear first
fpgaCtrl = 0x000c0000;    // Enable DCMs, use little-Endian format
FPGAWrite(FPGACtrlReg, 1, &fpgaCtrl);
Sleep(10);                // Wait 10 ms


// Start the DE interface
long deCtrl = 0;
FPGAWrite(DECtrlReg, 1, &deCtrl);         // Clear first
deCtrl = 0x80000000;
FPGAWrite(DECtrlReg, 1, &deCtrl);         // Start DE
// Wait until the DE clock is ready
bool ready = false;
long timeout = 100;  // With a 'Sleep' of 1 ms, timeout is 100 ms

while ((!ready) && (timeout > 0))
{
     timeout--;

     long fpgaStatus;
     FPGARead(FPGAStatusReg, 1, &fpgaStatus);
     ready = (fpgaStatus & 0x00100000) != 0;

     if (!ready)  Sleep(1);
}

// Check the value of timeout and report if there is a problem

// Clear the main control register
long mainCtrl = 0;
FPGAWrite(MainCtrlReg, 1, &mainCtrl);
```

6.3.2.2 Configuring/Starting Data Links

The serial data links are internally numbered from 0 to 1, or 0 to 11, depending on whether the *Standard* or the *Hi Rate* link option is present. The number of available data links can be retrieved from the instrument driver with the function:

```
long nbrLinks;
AcqrsD1_getInstrumentInfo(instrumentID, "LogDevDataLinks", &nbrLinks);
```

Each serial data link in use, numbered by *linkNbr* (0 .. n), must be configured and started explicitly. When started, the links run continuously at a data rate of 2.5 Gbits/s. Whenever there are no useful data (no payload), they transmit an *Idle* sign.

The following configuration code is required:

```
long linkCtrl = 0x003f00001;
if (nbrLinks == 2)              // Standard 2-fiber link
{
     if (linkNbr == 0)
          linkCtrl += 0x02000000;
     else
          linkCtrl += 0x03000000;
}

else                            // HiRate 12-fiber link
{
     if (linkNbr < 6)
          linkCtrl += 0x03000000;
}

long SLCbase = StreamerSLCbase1 + SLCbaseOffset*linkNbr;
FPGAWrite(SLCbase+SLCctrlOffset, 1, &linkCtrl);
```

The software must wait until the link is actually ready and running, with this code, for **each** link:

```
bool ready = false;
long timeout = 500;  // With 10 ms 'Sleep' --> 5 s timeout

while ((!ready) && (timeout > 0))
{
     long slcStatus;
     // Is the transmitter ready?
     FPGARead(SLCbase+SLCstatusOffset, 1, &slcStatus);
     if ( (slcStatus & 0x14) == 0x14)
          ready = true;
     if (!ready)
          Sleep(10);
     timeout--;
}
```

Here you should check that *timeout* is still positive, i.e. no timeout occurred.

If the data links have been successfully started, the 2 front panel LEDs next to the data links should turn green.

6.3.2.3 Configuring/Starting Data Transfers

The following configuration code is required:

```
// (A) Define the use of all data links with User Register 72
long linkConf = 0;
for (long link = (nbrLinks-1); link >= 0; link--)
{
     linkConf = linkConf<<2;
     linkConf += linkMode[link];
}
if (SC210)
     linkConf += 0x01000000;         // Enable only stream A
else
     linkConf += 0x05000000;         // Enable both streams A and B

linkConf += (streamMode<<30) | (headerType<<28);
FPGAWrite(StreamerConfigGlob, 1, &linkConf);
```

```
// (B) Define the frame sizes in User Registers 73/74
long streamConf = (1<<16) + (nbrSFsamples/16);

FPGAWrite(StreamerConfigSrcA, 1, &streamConf);
FPGAWrite(StreamerConfigSrcB, 1, &streamConf);


// (C) Load the SoftSync header values (even if they are not used)
FPGAWrite(StreamerSoftSyncW0, 1, &softSync[0]);
FPGAWrite(StreamerSoftSyncW1, 1, &softSync[1]);
FPGAWrite(StreamerSoftSyncW2, 1, &softSync[2]);
FPGAWrite(StreamerSoftSyncW3, 1, &softSync[3]);


// (D) Set Data Streams and Enable framing in the main control register
long mainCtrl;
FPGARead(MainCtrlReg, 1, &mainCtrl);
mainCtrl |= 0x140;
FPGAWrite(MainCtrlReg, 1, &mainCtrl);
```

Comments:

- The array **linkMode[nbrLinks]** must, for each link, contain a value between 0 and 2, defining the use of the serial data link: 0 = unused, 1 = connected to stream A, 2 = connected to stream B. It permits a completely free assignment of the data links to data streams.
- The variable **SC210** is used as a Boolean to distinguish between the 2 module versions
- The variable **streamMode** must take one of these values: 0 = Start on Trigger, 1 = Continuous, 2 = Triggered. These modes are further described in section **4.3.1, Data Streaming Mode**.
- The variable **headerType** must take on one of these values: 0 = no header, 1 = 16-byte header, 2 = 16-byte header + 16-byte SoftSync block.
- The variable **nbrSFsamples** is the number of 8-bit data samples in a stripe frame. It must be a multiple of 16 and must be in the range [2048, …. (16384-headerSize)], where headerSize = 0, 16, or 32.
- The array **softSync[4]** must contain the desired 128-bit SoftSync data pattern, if the **headerType** was specified as 2.
- The actual data transfer over the links is started by setting the framing bit in the main control register. Since this register already contains some necessary configuration bits, we read the contents of the main control register, set the bits, and write it back to the FPGA.

6.3.2.4 Monitoring the Data Transfer Operation

After the start of the streaming operation, the SC240/210 runs *without* any further software intervention. To monitor the progress, however, it is possible to sample the contents of two intermediate buffers.

A monitoring cycle consists of making the request for capturing data into one or two monitor buffers, waiting for the capture to be terminated, and reading the corresponding monitor buffers. The monitoring is always associated with a single link.

The following sample code assumes that both the DS and the TX-Monitor buffers are captured and read.

(1) Requesting Capture

```
// (A) Set up the 'Data Stream' Monitor Buffer with User Register 65
long bufConfDs = 0x20000000;    // Stream A (for Stream B use 0x20100000)
FPGAWrite(DsBufCtrl, 1, &bufConfDs);


// (B) Set up the 'Transfer' Monitor Buffer with User Register 66
long bufConfTx = 0x10000000;    // Stream A (for Stream B use 0x10100000)
bufConfTx += 0x10000 * linkToMonitor;
FPGAWrite(TxBufCtrl, 1, &bufConfTx);
```

```
// (C) Request the actual capture in the main control register
long mainCtrl;
FPGARead(MainCtrlReg, 1, &mainCtrl);
mainCtrl &= ~0x3000;      // Clear the capture bits (from previous op)
FPGAWrite(MainCtrlReg, 1, &mainCtrl);

mainCtrl += 0x1000;            // Request capture of DS monitor buffer
mainCtrl += 0x2000;            // Request capture of TX-Monitor buffer
FPGAWrite(MainCtrlReg, 1, &mainCtrl);
```

You must indicate the link number that you want to monitor, with the variable *linkToMonitor*. You also must indicate which data stream (A or B) this link is connected to. This must be the same as what was defined by the variables *linkMode[ ]* in the section **6.3.2.3**, **Configuring/Starting Data Transfers**.

This example code will capture the same data frame in both buffers only for the streaming modes "Start on Trigger" and "Continuous". In mode "Triggered", you will have to set both the four MSB of **bufConfIn** to 1 and the four MSB of **bufConfTx** to 2, to capture the same frame in both buffers.

(2) Waiting for End of Capture

```
// (A) Wait until User Register 65 shows 'DS' Monitor Buffer Ready
long statusDs;
FPGARead(DsBufCtrl, 1, &statusDs);
bool readyDs = (statusDs & 0x80000000) != 0;


// (B) Wait until User Register 66 shows TX-Monitor Buffer Ready
long statusTx;
FPGARead(TxBufCtrl, 1, &statusTx);
bool readyTx = (statusTx & 0x80000000) != 0;
```

You must loop over these statements until both 'readyRw' and 'readyTx' are true, or until you do too many iterations. Typically, the capture should occur in some tens of microseconds. Since the interrogation of the status register takes several microseconds, it is reasonable to set a limit of 100 iterations as a timeout.

(3) Reading Monitor Buffers

```
// (A) Reading the 'Ds' Monitor Buffer
long rwAddrOffset, startAddr;
FPGARead(DsBufCtrl,   1, &rwAddrOffset);
startAddr = rwAddrOffset&0x0000ffff;
FPGAWrite(StartAddrReg, 1, &startAddr);


long bufAddr = DsMonitorAddress;
FPGAWrite(IndirAddrReg, 1, &bufAddr);


long nbrLongs = nbrSamples/4;
long dataBuffer[nbrLongs];
FPGARead(ReadAddrReg, nbrLongs, &dataBuffer);


// (B) Reading the TX-Monitor Buffer
long startAddr = 0;
FPGAWrite(StartAddrReg, 1, &startAddr);


long bufAddr = TxMonitorAddress;
FPGAWrite(BufferIDReg, 1, &bufAddr);


long nbrLongs = nbrSamples/4;
long dataBuffer[nbrLongs];
FPGARead(ReadAddrReg, nbrLongs, &dataBuffer);
```

The value **nbrSamples** would typically be the number of samples per stripe frame, including the header size.

6.3.2.5 Stopping Data Transfers

The data transfers to the serial data links are stopped by clearing the 'framing' bit in the main control register.

// Disable framing in the main control register

```
long mainCtrl;
FPGARead(MainCtrlReg, 1, &mainCtrl);
mainCtrl &= 0xfffffeff;
FPGAWrite(MainCtrlReg, 1, &mainCtrl);
```

6.3.2.6 Stopping the Data Links

The data links are stopped by writing 0 into the control register of each used link *linkNbr* (0 – n).

```
long lnkCtrl = 0;
long SLCbase = StreamerSLCbase1 + SLCbaseOffset*linkNbr;
FPGAWrite(SLCbase+SLCctrlOffset, 1, &lnkCtrl);
```

6.4 Registers in Base and Streaming Firmware

Many of the registers listed below are only used in tests. For a description of the streaming registers, refer to section **6.4.7 *BASE STREAMER SPECIFIC REGISTERS*** and **6.4.7.1 *MAIN CONTROL REGISTER***. For FDK users needing a detailed description of the Base Test registers please consult the FDK Reference Manual, FDKUserManualACSC2x0.pdf.

6.4.1 Register List

			Firmware				
Register Number	**Register Address**	**Access Right**	**Base Test**	**Base Streaming**	**Standard(2) streaming**	**HRODL(12) streaming**	**Comment**
Customer Register Space – Reserved for Definition by Acqiris							
0	0x2200	RW	✓	✓	✓	✓	Indirect Access Port
1	0x2204	RW	✓	✓	✓	✓	FPGA Indirect Address
2	0x2208	RW	✓	✓	✓	✓	FPGA Buffer Identifier
3	0x220C	RW	✓	✓	✓	✓	FPGA Main Control
4	0x2210	R	✓	✓	✓	✓	FPGA Code Protection
5	0x2214						Reserved
6	0x2218	R	✓	✓	✓	✓	FPGA Main Status
7	0x221C		✓	✓	✓	✓	FPGA Temperature
8	0x2220	RW	✓	✓	✓	✓	FPGA DE-Bus Control
9	0x2224	RW	✓	✓	✓	✓	FPGA Direct Access Block
10-11	--						Reserved
12	0x2230	RW		✓	✓		Trigger Control. Not available for the SC210Stream2 firmware.
13	0x2234	RW		✓	✓		Trigger Control Status Lo. Not available for the SC210Stream2 firmware.
14	0x2238	RW		✓	✓		Trigger Control Status Hi. Not available for the SC210Stream2

			Firmware				
							firmware.
15	0x223C	RW		✓	✓		Trigger Control Delay. Not available for the SC210Stream2 firmware.
16-31	--						Reserved
32	0x2280	RW	✓	✓	✓	✓	Front Panel PIO Control
33	0x2284	RW	✓	✓	✓	✓	Front Panel DAC Control
34	0x2288	RW	✓	✓	✓	✓	Front Panel LED Control
35	0x228C						Reserved
36	0x2290	RW	✓	✓	✓	✓	Front Panel µDB-IO Control
37	0x2294	RW	✓	✓	✓	✓	Front Panel µDB-IO Output
38	0x2298	R	✓	✓	✓	✓	Front Panel µDB-IO Input
39-63							Reserved
Customer Register Space for the base test firmware							
64	0x2300	RW	✓				Base Tests Control
65	0x2304	R	✓				Base Tests Status
66-127	--		✓				Free
Customer Register Space for the Base streaming firmware							
64	0x2300	RW		✓			Main Control
65	0x2304			✓			Reserved
66	0x2308	RW		✓			TX Monitor Control and Status
67	0x230C	RW		✓			RX Monitor Control and Status
68-72				✓			Reserved
73	0x2324	RW		✓			Streamer Configuration
74-79	0x2328	RW		✓			Reserved
80-82	0x2340 0x2344 0x2348	RW		✓			SLC Control Link 0 SLC Status Link 0 SLC Signal Link 0
83-127				✓			Free
Customer Register Space for the streaming firmware							
64	0x2300	RW			✓	✓	Main Control
65	0x2304	RW			✓	✓	DS Monitor Control
66	0x2308	RW			✓	✓	TX Monitor Control
67	0x230C	RW			✓	✓	RX Monitor Control
68	0x2310	RO			✓	✓	Streamer Status A
69	0x2314	RO			✓	✓	Streamer Status B
70	0x2318	RO			✓	✓	Firmware ID
71	0x231C				✓	✓	
72	0x2320	RW			✓	✓	Streamer Global Configuration
73	0x2324	RW			✓	✓	Streamer Configuration A
74	0x2328	RW			✓	✓	Streamer Configuration B
75	0x232C				✓	✓	
76	0x2330	RW			✓	✓	Streamer SoftSync 1
77	0x2334	RW			✓	✓	Streamer SoftSync 2

			Firmware				
78	0x2338	RW			✓	✓	Streamer SoftSync 3
79	0x233C	RW			✓	✓	Streamer SoftSync 4

80-127	0x2340+4*i
	0x2344+4*i
	0x2348+4*i
		RW			✓	✓	SLC Control (LinK #i) , i [0:11]
							SLC Status (LinK #i) , i [0:11]
							SLC Signal (LinK #i) , i [0:11]
							I = [0,1] for Standard(2) streamer
							I = [0 to 11] for HRODL(12) streamer

6.4.2 Indirect Addressing

In order to conserve address space, large data buffers within the FPGA are not directly mapped to User Registers, but are accessed with an indirect addressing method.

Indirect addressing access must be performed as follows:

1. Write the Start Address within the buffer of interest into the FPGA Indirect Address Register
2. Write the Buffer Identifier value into the FPGA Buffer Identifier Register
3. Perform, on the Indirect Access Port, as many read or write operations as required

The Indirect Address mapping is listed below.

For FDK users needing a detailed description of the Indirect Addressing please consult the file FDKUserManualACSC2x0.pdf.

6.4.3 Indirect Addressing in Base Test Firmware

Buffer Identifier	Start Address Range	Access Right	Comment
Customer Register Space – Acqiris Reserved			
0x08	0x0 - 0x1FFC	RW	DE-Buffer, 8K bytes per channel
0x0C	0x0 - 0x1FFC	RW	DE-Monitor, 8K bytes per channel
0x00- 0x7F			Reserved for Acqiris
Customer Register Space – Customer Reserved			
0x80-0xFF			Reserved for Customer

6.4.4 Indirect Addressing in Base Streaming firmware

Buffer Identifier	Start Address Range	Access Right	Comment
Customer Register Space – Acqiris Reserved			
0x08	0x0 - 0x1FFC	RW	DE-Buffer, 8K samples per channel
0x10	0x0 - 0xFFFC	RW	TX-Monitor 64K bytes
0x20	0x0 - 0xFFFC	RW	RX-Monitor 64K bytes

6.4.5 Indirect Addressing in Data Streaming firmware

Buffer Identifier	Start Address Range	Access Right	Comment
Customer Register Space – Acqiris Reserved			
0x08	0x0 - 0x1FFC	RW	DE-Buffer, 8K samples per channel
0x0C	0x0 - 0x7FFC	RW	DS-Monitor 32K bytes
0x10	0x0 - 0x3FFC	RW	TX-Monitor 16K bytes
0x20	0x0 - 0x3FFC	RW	RX-Monitor 16K bytes

6.4.6 Base and Data Streamer Common Registers

6.4.6.1 Indirect Access Port

This register, together with the Indirect Address Register and the Buffer Identifier Register, gives access to large data blocks. As seen by the control software, it acts like a FIFO data port.

Register Space	Register Number	Register Address
Customer	**0**	**0x2200**

31..0
`IndirData`

[31..0]	**`IndirData`**	**`RW`**	Indirect Data value. Every read or write access uses the indirect address defined by the Indirect Address and Buffer Identifier registers.

6.4.6.2 Indirect Address Register

Register Space	Register Number	Register Address
Customer	**1**	**0x2204**

31..0
`IndirAddr`

[31..0]	**`IndirAddr`**	**`RW`**	This register defines the address for Indirect Access. It is used when accessing the Indirect Access Port. This address is defined in bytes and is auto incremented by 4 for each read or written word from/to the Indirect Access Port.

6.4.6.3 Buffer Identifier Register

Register Space	Register Number	Register Address
Customer	**2**	**0x2208**

31..0
`IndirCtr`

[31..0]	**`IndirCtr`**	**`RW`**	This register defines the target for indirect Access. Available Buffers are listed in the section 6.4.3, 0, and 6.4.5.

6.4.6.4 AcqirisControl Register

Register Space	Register Number	Register Address
Customer	**3**	**0x220C**

31..1	0
`--`	`IntEn`

[0]	**`IntEn`**	**`RW`**	Must be set to ‘1’ to enable the interrupt.
[1]	**`TrigEnSel`**	**`RW`**	Shall be 0 for user trigger
[8]	**`BigEndian`**	**`RW`**	Bit not used. Shall be 0
[23..16]	**`Enb_Dcm`**	**`RW`**	Shall be set to 0x0C in order to start the streamer clocks.

6.4.6.5 TempMonitor

Register Space	Register Number	Register Address
Customer	**7**	**0x221C**

31..16	15	14..13	12..0

--	TMPE	--	TMP_Monitor

[12..0]	**`TMP_Monitor`**	**`R`**	FPGA temperature when monitoring is enabled
[15]	**`TMPE`**	**`RW`**	FPGA Temperature Monitoring Enable

6.4.6.6 DEControl Register

Register Space	Register Number	Register Address
Customer	**8**	**0x2220**

31	30..0
`DeStart`	`--`

[30..0]	**`Reserved`**	**`R`**	Must be set to 0.
[31]	**`DeStart`**		When set to ‘1’, enables the data stream from the DE-Buffers to the FFT core.

6.4.6.7 Trigger Control Register

NOTE Users are only allowed to control the bit **`TRDI`** to reset the DCM (in order to get them locked), the bits **`TRM`** to set the trigger mode and the bit **`TRTC`** to reset the timestamp. All other bits shall remain ‘0’.

Register Space	Register Number	Register Address
Customer	**12**	**0x2230**

31	30..25	24
`TRPM`	`TRPH`	`TRPE`

23..19	18..8	7	3	2	0..1
`--`	`SDEL`	`ESDS`	`TRTC`	`TRDI`	`TRM`

[1..0]	**`TRM`**	**`RW`**	Trigger Mode	
			`00`	Low resolution trigger. The resolution is 16 samples when the channels are not interleaved or 32 samples in case of the SC240 in interleaved channel mode.
			`01`	High resolution trigger. The resolution is 1 sample when the channels are not interleaved or 2 samples in case of the SC240 in interleaved channel mode. The high resolution trigger is valid only for 1GS/s sampling.
			`10`	Automatic Asynchronous Trigger based on the local bus clock. For test purposes.
			`11`	Synchronous Trigger for test purposes. Needs the external clock and trigger generator.
[2]	**`TRDI`**	**`RW`**	Initialization of the DCM	
[3]	**`TRTC`**	**`RW`**	Enable Reset of the Timestamp Trigger counter to 0	
[7]	**`ESDS`**	**`RW`**	Enable Shift out of the synchronous delay	
[18..8]	**`SDEL`**	**`RW`**	Delay for synchronous trigger test. Needs the external board for clock and trigger generation.	
[24]	**`TRPE`**	**`RW`**	Enable Trigger fine phase correction:	
			`0`	Phase correction disabled
			`1`	Phase correction enabled
[30..25]	**`TRPH`**	**`RW`**	Initial phase value: unsigned 0 to 64	
[31]	**`TRPM`**	**`RW`**	Phase increment / decrement	
			`0`	Decrement phase relative to **`TRPH`**
			`1`	Increment phase relative to **`TRPH`**

6.4.6.8 Trigger Status Lo

This register is used to retrieve the trigger status and the lower part of the trigger timestamp value

Register Space	Register Number	Register Address
Customer	**13**	**0x2234**

31..8	7..4
TSTL	--

3	2	2	1
PDN	GRN	PDN	TRO

[0]	**TRO**	**R**	Trigger Occurred 0 No Trigger 1 A trigger has occurred
[1]	**PDN**	**R**	Set to '1' at the end of the DCM calibration phase
[2]	**GRN**	**R**	'1' indicates that the gray value of the trigger interpolator was not valid
[3]	**LCK**	**R**	'1' denotes that the trigger DCMs are locked
[31..8]	**TSTL**	**R**	Bit 23 to 0 of the Trigger Timestamp. The value is Valid if the bit **GRN** of the Register Trigger Status LO is '0' and the bit **TRO** of the Register Trigger Status LO is '1'.

6.4.6.9 Trigger Status Hi

This register is used to retrieve the upper part of the trigger Timestamp value.

Register Space	Register Number	Register Address
Customer	**14**	**0x2238**

31..0
TSTH

[31..0]	**TSTH**	**R**	Bit 55 to 24 of the Trigger Timestamp. The value is Valid if the bit **GRN** of the Register Trigger Status LO is '0' and the bit **TRO** of the Register Trigger Status LO is '1'.

6.4.6.10 Trigger Delay

This register could be used to delay the trigger event relative to the data. The resolution is one sample when the module operates in dual channel mode or two samples when the two channels are interlaced to double the sample rate.

Register Space	Register Number	Register Address
Customer	**15**	**0x223C**

31..8	6..0
	TRDL

[6..0]	**TRDL**	**RW**	Trigger Compensation. The Trigger event is shifted forward, relatively to the data stream, by a number of samples equal to the value of **TRDL**. The valid range is 0 to 127.

6.4.7 Base Streamer specific registers

The register definitions listed in this section are valid only for the Base Streaming firmware.

6.4.7.1 Main Control Register

This register defines the main control of the Base Streamer Application.

Register Space	Register Number	Register Address
Customer	**64**	**0x2300**

31..24	23..16
ACCN	-

15	14	13	12	11..10	9	8	7..6	5..4	3..0
BIDIR	CPRB	CPTB		-		STFX	-	-	-

[8]	**STFX**	**RW**	Start/Stop framing process as defined by the Streamer Mode 0 Stop 1 Start
[13]	**CPTB**	**RW**	Setting **CPTB** to '1' will start the capture of a new TX-Monitor buffer according to the configured Capture Mode. **CPTB** shall remain '1' until the TX-Monitor buffer is full unless the programmer wants to abort the capture by setting **CPTB** back to '0'.
[14]	**CPRB**	**RW**	Setting **CPRB** to '1' will start the capture of a new RX-Monitor buffer according to the configured Capture Mode. **CPRB** shall remain '1' until the RX-Monitor buffer is full unless the programmer wants to abort the capture by setting **CPRB** back to '0'.
[15]	**BIDIR**	**RW**	This bit shall be set '1' when the RX-Monitor buffer is used for monitoring.
[31..24]	**ACCN**	**RW**	Number of Accumulations before sending accumulated data. Values from 0 to 255 for number of accumulations 1 to 256.

To restart another capture, the start capture bits, CPTB and CPRB, shall be set to '0' and then set '1' again.

6.4.7.2 TX-Monitor Buffer Control and Status

Register Space	Register Number	Register Address
Customer	**66**	**0x2308**

31	30	29..28	27	26	25	24	23	22	21..20	19..16
RDY		-	- -						-	-

15..0
-

[31]	**RDY**	**R**	Buffer Ready: After a capture is started, **RDY** is set '1' when the TX-Monitor buffer is ready for readout. **RDY** is set '0' when the capture bit **CPTB** of the Main Control Register is set '0'.

6.4.7.3 RX-Monitor Buffer Control and Status

Register Space	Register Number	Register Address
Customer	**67**	**0x230C**

31	30	29..28	27	26	25	24	23	22	21..20	19..16
RDY		-	-						-	-

15..0
-

[31]	**RDY**	**R**	Buffer Ready: After a capture is started, **RDY** is set '1' when the RX-Monitor buffer is ready for readout. **RDY** is set '0' when the capture bit **CPTB** of the Main Control Register is set '0'.

6.4.7.4 Base Streamer Configuration Register

Register 73 defines the length of the Stripe Frame for Streaming.

Register Space	Register Number	Register Address
Customer	**73**	**0x2324**

31..16
-

15..0
SFS

[15..0]	**SFS**	**RW**	Size of a Stripe Frame in units of 16-sample blocks. **SFS** does not take into account the size of the Header. The number of samples transmitted with a Stripe Frame is **SFS** x 16 plus the Header size. The valid range is 256 samples to 64K samples (**SFS** = 0xF to 0xFFF)

6.4.8 Data Streamer specific registers

The registers listed in this section are valid only for the Data Streaming firmware.

6.4.8.1 Main Control Register

This register defines the main control of the Streamer Application.

Register Space	Register Number	Register Address
Customer	**64**	**0x2300**

31..16
-

15	14	13	12	11..10	9	8	7..6	5..4	3..0
-	CPRB	CPTB	CPDB	-	STRX	STFX	CSTB	CSTA	-

[5..4] **CSTA** **RW** **Configure Stream A:** Data Source Selection

00	Channel A
01	Channel B
10	Unused
11	Test

[7..6] **CSTB** **RW** **Configure Stream B:** Data Source Selection

00	Channel A
01	Channel B
10	Unused
11	Test

[8] **STFX** **RW** Start/Stop framing process as defined by the selected Streamer Mode

0	Stop
1	Start

[9] **STRX** **RW** Start/Stop framing reference data [For Production Test Only].

0	Stop
1	Start

[12] **CPDB** **RW** Setting **CPDB** to ‘1’ will start the capture of a new DS-Monitor buffer according to the configured Capture Mode. **CPDB** shall remain ‘1’ until the DS-Monitor buffer is full unless the programmer wants to abort the capture by setting **CPDB** back to ‘0’.

[13] **CPTB** **RW** Setting **CPTB** to ‘1’ will start the capture of a new TX-Monitor buffer according to the configured Capture Mode. **CPTB** shall remain ‘1’ until the TX-Monitor buffer is full unless the programmer wants to abort the capture by setting **CPTB** back to ‘0’.

[14]	**CPRB**	**RW**	Setting **CPRB** to '1' will start capture a new RX-Monitor buffer according to the configured Capture Mode. **CPRB** shall remain '1' until the RX-Monitor buffer is full unless the programmer wants to abort the capture by setting **CPRB** back to '0'.

To restart another capture, the start capture bits, CPDB – CPTB and CPRB shall be set to '0' and then set '1' again.

6.4.8.2 DS-Monitor Buffer Control & Status

This register controls the In-buffer that enables capturing input data before the framing process.

Register Space	Register Number	Register Address
Customer	**65**	**0x2304**

31	30	29..28	27	26	25	24	23	22	21..20	19	18	17	16
RDY	BF	CMD	--						STID	--			

15..0
LWADD

[15..0]	**LWADD**	**R**	The value indicates the position within the DS-Monitor buffer of the first sample acquired with the last successful capture. The value of **LWADD** shall be used to configure the FPGA Indirect Address Register before reading the DS-Monitor buffer.	
[20..21]	**STID**	**RW**	Stream ID: select the stream to monitor with the DS-Monitor buffer. Please note there is no coherency checking between the capture mode requested and the selected stream configuration.	
			00	Stream A
			01	Stream B
[28..29]	**CMD**	**RW**	Select the capture mode.	
			00	Immediate: Capture start immediately after the bit **CPDB** of the Main Control Register is set '1'
			01	Triggered: Capture start after the first trigger after the bit **CPDB** of the Main Control Register is set '1'
			10	TX Triggered: Capture start synchronized by the TX-Monitor capture after the bit **CPDB** of the Main Control Register is set.
			11	Reserved
[30]	**BF**	**R**	Buffer Full status: After a capture is started, **BF** is set '1' when the DS-Monitor buffer is full. **BF** is set '0' when the capture bit **CPDB** of the Main Control Register becomes '0'.	
[31]	**RDY**	**R**	Buffer Ready: After a capture is started, **RDY** is set '1' when the DS-Monitor buffer is ready for readout. **RDY** is set '0' when the capture bit **CPDB** of the Main Control Register becomes '0'.	

6.4.8.3 TX-Monitor Buffer Control & Status

This register controls the TX Stripe Frame Monitor Buffer that enables capturing a Stripe Frame on any active data link.

Register Space	Register Number	Register Address
Customer	**66**	**0x2308**

31	30	29..28	27	26	25	24	23	22	21..20	19..16
RDY		CMD	--						STID	LKID

15..0
LWADD

[15..0]	**LWADD**	**R**	Last Write Address. This value is related to the address of the sample block last written.

[19..16]	**LKID**	**RW**	Link ID: it defines the link source for the next Stripe frame capture. Must be in the range 0–1 for the standard streamer and 0-11 for the 12 optical link option.
[20..21]	**STID**	**RW**	Stream ID: select the stream to monitor with the TX-Monitor buffer. Please note there is no coherency checking between the capture mode requested and the selected stream configuration.

00	Stream A
01	Stream B

[28..29]	**CMD**	**RW**	Select the capture mode.

00	Immediate: Capture the next TX Stripe Frame on the selected stream **STID** and selected Link **LKID.** Capture starts immediately after the bit **CPTB** of the Main Control Register is set.
01	Link Triggered: Capture the next TX Stripe Frame on the selected stream **STID** and selected Link **LKID.** Capture starts immediately after the bit **CPTB** of the Main Control Register is set '1'. This is the value to set for 'Continuous' and 'StartOn Trig' streaming mode.
10	Triggered: Capture the next TX Stripe Frame on the selected stream **STID** and selected Link **LKID** occurring after the trigger. This is the value to set for the Triggered streaming operation.
11	Reserved

[31]	**RDY**	**R**	Buffer Ready: After a capture is started, **RDY** is set '1' when the TX-Monitor buffer is ready for readout. **RDY** is set '0' when the capture bit **CPTB** of the Main Control Register is set '0'.

Please note that the address of the first sample block of the TX Stripe Frame Monitor buffer is always 0x0. The FPGA Indirect Address Register shall be set to 0x0 before reading the TX-Monitor buffer.

6.4.8.4 RX-Monitor Buffer Control & Status

This register controls the RX Stripe Frame Buffer that enables capturing a Stripe Frame on the receiver interface of each data link controller. Please note that the corresponding data link controller should be configured in bidirectional mode.

Register Space	Register Number	Register Address
Customer	**67**	**0x230C**

31	30	29..28	27	26	25	24	23	22	21..20	19..16
RDY		CMD	--						STID	LKID

15..0
LWADD

[15..0]	**LWADD**	**R**	Last Write Address. This value is related to the address of the sample block last written.
[19..16]	**LKID**	**RW**	Link ID: it defines the link source for the next Stripe frame capture. Must be in the range 0–1 for the standard streamer and 0-11 for the 12 optical link option.
[20..21]	**STID**	**RW**	Stream ID: select the stream to monitor with the RX-Monitor buffer. Please note there is no coherency checking between the capture mode requested and the selected stream configuration.

00	Stream A
01	Stream B

[28..29]	**CMD**	**RW**	Select the capture mode.

00	Immediate: Capture the next RX Stripe Frame on the selected stream **STID** and selected Link **LKID.** Capture starts immediately after the bit **CPRB** of the Main Control Register is

set ‘1’.

			`01`	Link Triggered: Capture the next RX Stripe Frame on the selected stream **STID** and selected Link **LKID.** Capture starts immediately after the bit **CPRB** of the Main Control Register is set ‘1’. This is the value to set for ‘Continuous’ and ‘StartOn Trig’ streaming mode.
			`10`	Tx Triggered: Capture the next RX Stripe Frame on the selected stream **STID** and selected Link **LKID** synchronized by the TX-Monitor capture. This is for automatic checking purpose.
			`11`	Reserved
[31]	**RDY**	**R**	Buffer Ready: After a capture is started, **RDY** is set ‘1’ when the RX-Monitor buffer is ready for readout. **RDY** is set ‘0’ when the capture bit **CPRB** of the Main Control Register is set ‘0’.	

Please note that the address of the first sample block of the RX Stripe Frame Monitor buffer is always 0x0. The FPGA Indirect Address Register shall be set to 0x0 before reading the RX-Monitor buffer.

6.4.8.5 Streamer Global Status Register

This register contains the high level status of each stream and link.

Register Space	Register Number	Register Address
Customer	**68**	**0x2310**

31	30	29..27	26..24	23..22	21..20	19..18	17..16
SSTA1	SSTA0	SERRB	SERRA	LK11S	LK10S	LK9S	LK8S

15..14	13..12	11..10	9..8	7..6	5..4	3..2	1..0
LK7S	LK6S	LK5S	LK4S	LK3S	LK2S	LK1S	LK0S

[2*i+1:2*i]	**LKiS**	**R**	Link i Status, I in the range 0-1 for the Standard(2) streamer and in the range 0-11 for the HRODL(12) option.	
			`00`	Not implemented
			`01`	Ready to frame
			`10`	Frame - Paused
			`11`	Frame - Transmitting
[26..24]	**SERRA**	**R**	Data Stream A Error Code. It’s a copy of the 3 MSB of the Streamer A Error code **ERCD** of the streamer A status register.	
[29..27]	**SERRB**	**R**	Data Stream B Error Code. It’s a copy of the 3 MSB of the Streamer B Error code **ERCD** of the streamer B status register.	
[30]	**SSTA0**	**R**	Streamer Error: at least one error occurred within one data stream. It is ‘1’ when at least one of the error code **ERCD** of the stream A or B status register indicates an error.	
[31]	**SSTA1**	**R**	Streamer Running: ‘1’ indicates at least one data streamer is running. It is a copy of the bit **STFX** of the Main Control register.	

6.4.8.6 Streamer A/B Status Registers

The register 69 defines the status of stream A, and register 70 that of stream B.

Register Space	Register Number	Register Address
Customer	**69 / 70**	**0x2314 / 0x2318**

31..28	27..24	23..22	21..18	17..16
ERCD	STCD	--	ALNK	EOB(17..16)

15..0
EOB(15..0)

[17..0]	**EOB**	**R**	Effective Data Bandwidth on Stream **A/B**

[21..18]	**ALNK**	**R**	Link ID of the currently active link on stream **A/B**
[27..24]	**STCD**	**R**	Streamer **A/B** Status Code

24 ‘1’ when the streamer is initialized and transfers valid data to the link, otherwise ‘0’.

25 ‘1’ when the streamer is waiting for a trigger while the streaming mode is set in triggered mode with **SMOD** of the streamer global control register set to ‘10’.

26 ‘1’ when ongoing read does not complete while the streamer **A/B** has been stopped by setting the bit **STFX** of the Main control register to ‘0’.

27 --

[31..28] **ERCD** **R** Streamer **A/B** Error Code

28 Header not inserted

29 Ongoing read not complete

30 --

31 Overflow, next link not ready

6.4.8.7 Streamer Global Configuration Register

It defines the parameters shared by the streamer A and B and the association of the links to one of the two streams A or B.

Register Space	Register Number	Register Address
Customer	**72**	**0x2320**

31..30	29..28	27..26	25..24	23..22	21..20	19..18	17..16
SMOD	HTYP	SCFGB	SCFGA	LK11M	LK10M	LK9M	LK8M

15..14	13..12	11..10	9..8	7..6	5..4	3..2	1..0
LK7M	LK6M	LK5M	LK4M	LK3M	LK2M	LK1M	LK0M

[2*i+1:2*i] **LKiM** **RW** Link i Scheduling Mode, i in the range 0-1 for the Standard(2) streamer and in the range 0-11 for the HRODL(12) option.

00 Not implemented

01 Linked to stream A

10 Linked to stream B

11 --

[25..24] **SCFGA** **RW** Streamer A Configuration

00 Disable

01 Enable

10,11 --

[27..26] **SCFGB** **RW** Streamer B Configuration

00 Disable

01 Enable

10,11 --

[29..28] **HTYP** **RW** Header Type

00 No header.

01 Header without SoftSync pattern

10 Header with SoftSync pattern

11 --

[31..30] **SMOD** **RW** Streaming mode.

00 Continuous, start on trigger.

01 Continuous, start immediately

`10`	Triggered
`11`	--

6.4.8.8 Streamer A / B Configuration Registers

Register 73 defines the length of the Stripe Frame for Stream A, Register 74 for Stream B.

Register Space	Register Number	Register Address
Customer	**73 / 74**	**0x2324 / 0x2328**

31..16
`UFS`

15..0
`SFS`

[15..0]	**`SFS`**	**`RW`**	Size of a Stripe Frame for Stream **A / B** in units of 16-sample blocks. **`SFS`** does not take into account the size of the Header. The number of samples transmitted with a Stripe Frame is **`SFS`** x 16 minus the Header size in count of bytes.
[31..16]	**`UFS`**	**`RW`**	Must be set to 1.

6.4.8.9 Streamer Soft Sync Registers

These 4 registers define the SoftSync pattern to be embedded as a part of the Header field of the Stripe Frame, depending on the selected frame header format.

Register Space	Register Number	Register Address
Customer	**76 / 77 / 78 / 79**	**0x2330 / 0x2334 / 0x2338 / 0x233C**

31..24	23..16
`SSY_B0`	`SSY_B1`

15..8	7..0
`SSY_B2`	`SSY_B3`

[7..0]	**`SSY_B3`**	**`RW`**	SoftSync Byte 3, 7, 11, 15 (in registers 76, 77, 78, 79 respectively)
[15..8]	**`SSY_B2`**	**`RW`**	SoftSync Byte 2, 6, 10, 14 (in registers 76, 77, 78, 79 respectively)
[23..16]	**`SSY_B1`**	**`RW`**	SoftSync Byte 1, 5, 9, 13 (in registers 76, 77, 78, 79 respectively)
[31..24]	**`SSY_B0`**	**`RW`**	SoftSync Byte 0, 4, 8, 12 (in registers 76, 77, 78, 79 respectively)

6.4.8.10 SLC Control Registers

The SLC Control registers define the configuration and running modes of the Serial Front Panel Data Port Controller. There is one control register for each physical data link, *LnkNbr* = 0 – 11.

Register Space	Register Number	Register Address
Customer	**80 + 4 * LnkNbr**	**0x2340 + 0x10 * LnkNbr**

Where LnkNbr is 0-1 for the Standard(2) streamer and 0-11 for the HRODL(12) option.

31	30	29-26	25	24	23..16
RSF	RRX	--	RXP	TXP	RXFT

15..14	13	12	11..9	8	7	6	5..4	3	2	1	0
--	FSY	SYF	--	RFC	CRC	MST	--	FWC	CPY	RXE	TXE

[0]	**TXE**	**RW**	TX Controller Enable
			0 Disabled 1 Enabled
[1]	**RXE**	**RW**	RX Controller Enable
			0 Disabled 1 Enabled
[2]	**CPA**	**RW**	TX Copy Mode. **TXE** must be enabled.
			0 Disabled 1 Enabled
[3]	**FWC**	**RW**	TX Flow Control. **TXE** and **RXE** must be enabled.
			0 Disabled 1 Enabled
[5..4]	**Res**	**RW**	Reserved, shall be set to zero
[6]	**MST**	**RW**	TX Copy Master Mode. **TXE** and **CPY** must be enabled.
			0 Disabled 1 Enabled
[7]	**CRC**	**RW**	CRC Encoding Enable
			0 Disabled 1 Enabled
[8]	**RFC**	**RW**	RX Flow Control. **TXE** and **RXE** must be enabled.
			0 Disabled 1 Enabled
[11..9]	**Res**	**RW**	Reserved, shall be set to zero
[12]	**SYF**	**RW**	Mark TX Frame with SYNC without data
			0 Disabled 1 Enabled
[13]	**FSY**	**W**	Send a SYNC without data frame
			0 Disabled 1 Enabled
[23..16]	**RXFT**	**RW**	RX FIFO Threshold for Flow Control
[24]	**TXP**	**RW**	TX Polarity. 0 = Default Polarity 1 = Invert Polarity
[25]	**RXP**	**RW**	RX Polarity. 0 = Default Polarity 1 = Invert Polarity
[29..26]	**Debug**	**RW**	Shall be set to zero
[30]	**RRX**	**W**	Reset RX Controller. 1 = Reset Flags 0 = Default.
[31]	**RSF**	**RW**	Reset Status Flags. 1 = Reset Flags 0 = Default.

6.4.8.11 SLC Status Registers

The SLC Status registers indicate the current status of the Serial Front Panel Data Port Controller. There is one status register for each physical data link, *lnkNbr* = 0 – 11.

Register Space	Register Number	Register Address
Customer	**81 + 4 * LnkNbr**	**0x2344 + 0x10 * LnkNbr**

Where LnkNbr is 0-1 for the Standard(2) streamer and 0-11 for the HRODL(12) option.

31..16
TXTGH

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RES	CRE	RDE	DER	FTE	SYE	--	EER	--	RXK	--	TXK	RXR	TXR	RXL	TXF

[0]	**TXF**	**R**	1 = TX Fault (Optical Transceiver Not Ready or Faulty)
[1]	**RXL**	**R**	1 = RX Loss of Signal (Optical Transceiver Not Ready or Faulty)
[2]	**TXR**	**R**	TX Physical Layer Ready. 0 = Not initialized or faulty 1 = Ready
[3]	**RXR**	**R**	RX Physical Layer Ready. 0 = Not initialized or faulty 1 = Ready
[4]	**TXK**	**R**	TX Link Layer Ready. 0 = Not initialized or faulty 1 = Ready
[6]	**RXK**	**R**	RX Link Layer Ready. 0 = Not initialized or faulty 1 = Ready
[8]	**EER**	**RW**	TX Encoding Error. 0 = No error 1 = Error Detected
[10]	**SYE**	**RW**	RX Loss of Sync Error. 0 = No error 1 = Error Detected
[11]	**FTE**	**RW**	RX Format Error. 0 = No error 1 = Error Detected
[12]	**DER**	**RW**	RX Data Encoding Error. 0 = No error 1 = Error Detected
[13]	**RDE**	**RW**	RX Running Disparity Error. 0 = No error 1 = Error Detected
[14]	**CRE**	**RW**	RX CRC Checking Error if relevant. 0 = No error 1 = Error Detected
[15]	**RES**	**RW**	reserved
[31..16]	**TXTGH**	**RW**	TX Effective Throughput (Relevant only if TX is Enabled)

6.4.8.12 SLC Signal Status register

Status of signal received or to be transmitted over the serial link.

Register Space	Register Number	Register Address
Customer	**82 + 4 * LnkNbr**	**0x2348 + 0x10 * LnkNbr**

Where LnkNbr is 0-1 for the Standard(2) streamer and 0-11 for the HRODL(12) option.

31..9	8
-	RX_FOVF

7	6	5	4	3	2	1	0
RX_NRDY	RX_DIR	RX_PIO2	RX_PIO1	TX_NRDY	TX_DIR	TX_PIO2	TX_PIO1

[0]	**TX_PIO1**	**RW**	Value of the PIO1 signal to be transmitted over the Serial Link
[1]	**TX_PIO2**	**RW**	Value of the PIO2 signal to be transmitted over the Serial Link
[2]	**TX_DIR**	**RW**	Value of the DIR signal to be transmitted over the Serial Link
[3]	**TX_NRDY**	**RO**	Value of the NRDY signal received from the Serial Link
[4]	**RX_PIO1**	**RO**	Value of the PIO1 signal received from the Serial Link
[5]	**RX_PIO2**	**RO**	Value of the PIO2 signal received from the Serial Link
[6]	**RX_DIR**	**RO**	Value of the DIRY signal received from the Serial Link
[7]	**RX_NRDY**	**RW**	Value of the NRDY signal to be transmitted over the Serial Link
[8]	**RX_FOVF**	**RO**	Value of the RX_FOVF signal received from the Serial Link