EUROPEAN SOUTHERN OBSERVATORY

Organisation Européenne pour des Recherches Astronomiques dans l'Hémisphère Austral

Europäische Organisation für astronomische Forschung in der südlichen Hemisphäre

VERY LARGE TELESCOPE

FIERA CCD Controller

Software

User Manual

Doc.No. VLT-MAN-ESO-13640-1388

Issue 3.0

Date 31/03/04

Prepared	C. Cumani, A. Balestra,	31/03/04	.....
	Name	Date	Signature
Approved	D. Baade		.....
	Name	Date	Signature
Released	G. Monnet		.....
	Name	Date	Signature

Change Record

Issue/Rev.	Date	Section/Page affected	Reason/Initiation/Document/Remarks
1.0	13/11/97	All	First issue
1.1	30/09/98	2.3.2 2.6, 2.13.2, 2.13.4 2.10, 2.13.2 2.8.1, 2.8.2 2.14 2.17 3 3.3 3.8 3.9 4 5 7.8.9	Added Multi Step exposures Added Centroiding and IP Functions Added Telemetry Added FIERA SPARC Powering Down and Rebooting procedures Proposed “image extension per port” format added Example of FIERA SW example modified Removed bodies of Include and Database Class files (included the references) Man pages added FITS extension example added Log file example added Installation and SLCU configuration widely updated New Chapter added, with manpages of utilities for FIERA checking Telemetry monotoring added
1.2	22/11/99	2.6, 2.14.2 2.14.2 2.17 New Chapter 3 5.6.3 5.6.5 5.7 5.7.1.4 5.7.6	Added description of average pixel value calculation Added description of fcdCEN_STANDARD_STR string New examples added added (for AO systems) RtapEnvTable usage corrected Precised usage of fcdInstall.sh Description of System files updated Description of xntp updated Changes for CCSLite
1.3	17/02/01	2.14.4 5 6	Added new public database attributes Installation procedures updated fcdWizard manpage removed
2.0	31/03/03	2.14.4 5.1	Added new public database attributes (CRPIX) SLCU Hardware configuration

Change Record

Issue/Rev.	Date	Section/Page affected	Reason/Initiation/Document/Remarks
3.0	31/03/04	1.4 5.6.3.1 5.7.1.1 5.7.2 5.7.5 5.7.6 5.8.1	Reference corrected No more RTAP on Instrument Workstation Note on ATM Interface name added CCDVENV environment variable added Setting of ACC_HOST corrected NOTE on `"logManager -f"` option Note on DSP driver added

1 INTRODUCTION

The software described in this manual is intended to be used in the ESO VLT project by ESO and authorized external contractors only.

While every precaution has been taken in the development of the software and in the preparation of this documentation, ESO assumes no responsibility for errors or omissions, or for damage resulting from the use of the software or of the information contained herein.

1.1 Purpose

This document is the User Manual of the FIERA CCD control software (all the modules with the **fcd**[1] prefix).
It is intended to provide people, who intend to use the FIERA CCD Controller, with all the necessary information to:

1. **install** from scratch the FIERA CCD software (section *Installation Guide*)
   Pre-requisites:
   a. Experience with the installation of other VLT sw packages.
   b. Knowledge of UNIX operating system.
   c. Knowledge of HP RTAP database.
   d. Familiarity with the VLT sw standard environment.
2. **interact programmatically** with the FIERA DCS sw (section *User's Guide*)
   Pre-requisites:
   a. Experience with programming within the VLT sw environment.
3. **operate** the CCD camera as simple **standalone** instrument (section *Standalone Operations*)
   Pre-requisites:
   a. Some experience with CCD camera operation.

The manual assumes that the reader has some knowledge of C/C++ and Tcl/Tk languages, UNIX Operating System, HP RTAP and the VLT Software, in particular CCS. It is not intended to be an introduction to CCD cameras, and therefore it uses common terminology in this field (e.g. pixel, binning, readout, frame-transfer chip, etc.) without further explanation.

In addition to the **Introduction**, this manual contains other major chapters:

User's Guide: it contains information about:

4. Definitions: exposure id, type and status, operational mode and state.
5. Image processing
6. Startup/Shutdown
7. Include files
8. Command Interface to the FIERA DCS software
9. Function Interface (C library) to the FIERA DCS software
10. GUI panel classes for applications using the FIERA DCS software.

1. The **fcd** prefix has been chosen to resemble the prefix (**ccd**) used for the LCU-ACE CCD Controller software modules.

11. Data
    a. Dictionary
    b. Setup files and setup parameters.
    c. Image files. Interface with Observation Software.
    d. Public section of the FIERA DCS on-line database.
12. Example of usage of the FIERA DCS Software from applications using scientific CCD cameras.

Reference: manual pages of functions available to external software, as well as scripts and processes interacting with external sw and their Command Definition Tables, include files and examples of files of various type to be used as templates.

Installation Guide: step-by-step description of the procedure to be followed to install the software.

Error Messages: reference to files containing information about errors produced by the FIERA CCD software.

Standalone Operation: description of the usage of the FIERA CCD Controller as a simple standalone instrument.

1.2 Scope

The present document is intended to be used for scientific CCD cameras.

1.3 Applicable Documents

The following documents, of the exact issue shown, form a part of this document to the extent specified herein. In the event of conflict between the documents referenced herein and the contents of this document, the contents of this document shall be considered as a superseding requirement.

[1] VLT-PRO-ESO-10000-0228, 1.0 10/03/93 ---- VLT Software Programming Standards

[2] VLT-SPE-ESO-17212-0001, 2.0 12/04/95 ---- VLT Instrumentation Sw Specification

[3] VLT-SPE-ESO-17240-0385, 2.1 15/07/96 ---- INS Common Software Specification

[4] GEN-SPE-ESO-19400-0794, 1.08 15/11/96 ---- ESO Data Interface Control Document

[5] VLT-SPE-ESO-13640-1266, 1.0 16/04/97 ---- FIERA CCD Controller, Software Functional Specifications

[6] VLT-ICD-ESO-17240-19400, 2.4 06/11/96 ---- ICD between VCS and VLT Archive System

1.4 Reference Documents

The following documents contain additional information and are referenced in the text.
The document versions are the ones defined in the VLT Common Software distribution to which the present document belongs.

[7] VLT-MAN-ESO-17200-0642 ---- VLT Common Software Installation Manual

[8] VLT-MAN-ESO-17200-2238 ----VLT Common Software, Combined OS Installation Manual

[9] VLT-MAN-ESO-17200-0888 ---- VLT Common Software Overview

[10] VLT-MAN-ESO-17210-0619 ---- VLT CCS User Manual

[11] VLT-MAN-ESO-17210-0707 ---- On Line Database Loader User Manual

[12]VLT-MAN-ESO-17210-0690 ---- Graphical User Interface User Manual
[13]VLT-MAN-ESO-17240-0637 ---- INS Common Sw - dxf User Manual
[14]VLT-MAN-ESO-17240-0726 ---- INS Common Sw - slx User Manual
[15]VLT-MAN-ESO-17240-0853 ---- INS Common Sw - oslx User Manual
[16]VLT-MAN-ESO-17240-0866 ---- INS Common Sw - rtd User Manual
[17]VLT-MAN-ESO-17240-0725 ---- INS Common Sw - pco User Manual
[18]VLT-MAN-ESO-17240-0672 ---- CCD Detector Control Sw - User Manual
[19]VLT-MAN-ESO-13640-1707 ---- FIERA Sw Maintenance Manual

1.5 Abbreviations and Acronyms

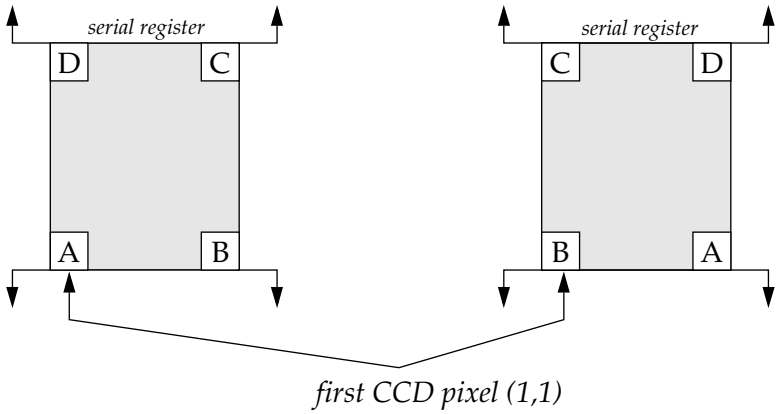
The following abbreviations and acronyms are used in this document:

ACE	Array Control Electronics
AO	Adaptive Optics
ATM	Asynchronous Transfer Mode
CCD	Charge-Coupled Device
CCS	Central Control Software
CDT	Command Definition Table
CPU	Central Processing Unit
DCL	Device Control Library
DCS	Detector Control Software
DMD	Data Management Division
ESO	European Southern Observatory
FDDI	Fiber Distributed Data Interface
FIERA	Fast Imager Electronic Readout Assembly
FITS	Flexible Image Transport System
HW	Hardware
INS	Instrumentation Software Package
I/O	Input/Output
LAN	Local Area Network
LCU	Local Control Unit
MIDAS	Munich Image Data Analysis System
N/A	Not Applicable
OLDB	Online Database
RMS	Root Mean Square
RTAP	Real Time Application Program
RTD	Real Time Display
SCCD	Scientific CCD
SLCU	SPARC Local Control Unit

SPARC	Scalable Processor Architecture
SW	Software
TBC	To Be Clarified
TBD	To Be Defined
TCS	Telescope Control Software
TIM	Time Interface Module
TRS	Time Reference System
UIF	(Portable) User Interface (Toolkit)
VLT	Very Large Telescope
WAN	Wide Area Network
WS	Workstation

1.6 Glossary

*First CCD pixel*With the following assumptions:
1)looking at a CCD from the illuminated side,
2)the serial registers define the horizontal direction,
3)the output register defined by the manufacturer as the first output (usually called A), is set at the bottom,
the *first pixel of a CCD* is the pixel on the bottom, left corner. It has coordinates 1,1.



From [5]:
*Overlapping windows*windows with pixels in common.

1.7 Stylistic Conventions

The following styles are used:

bold in the text, for commands, file names, etc. as they must be typed.

italic in the text, for parts that have to be substituted with the real content before typing.

`teletype` for examples.

<name> in the examples, for parts that have to be substituted with the real content before typing.

The **bold** and *italic* styles are also used to highlight words.

1.8 Naming conventions

This implementation follows the naming conventions as outlined in [2].

1.9 Problem Reporting/Change Request

The form described in [7] shall be used.

2 USER'S GUIDE

This part of the document provides a description of the programmatic interface of the FIERA CCD sw. It gives software engineers, writing applications using a FIERA CCD controller, information on what they have to implement in their code to be able to interact with the FIERA CCD sw.

For people having no experience with the FIERA CCD sw, it is highly recommendable, although not necessary, before starting to read this section, to first look at the interactive usage of the FIERA CCD sw as stand-alone instrument (see Chapter 8) and if possible to try it out. This exercise can help you get a better idea of the way how the FIERA CCD sw works and its functionality.

2.1 Overview

The FIERA CCD Detectors Control Software is distributed over two hardware platforms: WS and the FIERA " EmbeddedComputer" or SLCU (SPARC based LCU), with a SPARC CPU and two TMS320C40 DSPs (see [5]).

For a description of the hardware and software needed to run the FIERA CCD sw, see section 5.2.

A program accesses the functionality provided by the FIERA CCD sw in the following way:

1. It retrieves information about configuration and status of the CCD camera through functions available in the library *libfcd* (see section 2.12). Whenever the information wanted cannot be retrieved by means of library function calls, a direct access to the public part of the on-line database is possible (see section 2.14.4).
2. It asks the FIERA CCD sw for services by means of commands using the CCS message system (see section 2.11).
3. It can use the CCS event system to be informed about changes of status of the FIERA CCD system (see section 2.17).

The resulting image data can be returned as a FITS file format and/or as raw data to be shown through an online display.

Note: In the examples shown in the next sections we use the following setting:

```
setenv CCDNAME myccd
setenv RTAPENV wmyccd
setenv CCDLENV wslcu
```

2.2 Parallelism

One of the main requirements for the FIERA CCD sw is to optimize observation time, performing operations in parallel as much as possible.
For this reason, the portion of the image which has already been read out is transferred to the Workstation and saved in FITS file and/or displayed (depending from the setup), while the next image portion is being read out. This parallelism of readout and image transfer ensures that the time interval between the end of the readout and the completion of image display or FITS file on disk is minimized. For most applications using CCDs this level of parallelism is fully sufficient to fulfill the performance requirements: most applications will only be concerned with one active exposure at any given time.
On the other hand, some applications (e.g., very large CCDs) may need to push optimization further and therefore control more than one exposure at a given time (e.g., next integration is

started as soon the readout is completed, but the last portion of the image is still being transferred to the Workstation).
It has however to be remembered that this further optimization introduces complexity in the application: pro (better performance) and contra (complexity) should therefore be carefully evaluated.
Since more than one exposure can be active at a given point in time, the FIERA CCD sw provides several identical points in the on-line database, where information about one running exposure is stored (in the current release points *exposures:exposure_1* and *exposures:exposure_2* are provided).
Whenever a new exposure is started, and no exposure is already running, the point *exposures:exposure_1* is always selected by the FIERA CCD sw to store information related to that exposure.
Applications not handling parallel exposures must therefore always look for information in *exposures:exposure_1* and can ignore all other identical points.

2.3 Exposure definitions

2.3.1 CCD Exposure Id

In order to be able to uniquely identify an exposure among those already finished and those running, an identification number is associated to each exposure.

The exposure Id used by the FIERA CCD sw for this purpose is an integer sequential number (vltINT32).
The exposure Id can be:

- *either* passed to the FIERA CCD DCS software as a parameter of the command `START` (as defined in the INS Common Software document [3]),
- *or* generated by the FIERA CCD DCS software, in case the command `START` had no parameter (this in order to keep compatibility with the previous LCU based ESO CCD Control Software, see [18]).

The exposure Id is always returned as reply parameter to the command `START`.

Applications handling more than one running exposure at a time (see 2.2) must pass this Id as parameter to all commands dealing with a running exposure (e.g. `ABORT`, see section 4.2.1).
Special values (see *fcd.h*) can be used; applications are encouraged to use them whenever possible:

1. `fcdEXP_NEXT` refers to the next exposure to be started (typically for command `SETUP`).
2. `fcdEXP_LAST` refers to the last exposure started.

As described in [18], if the OS does not pass the exposure Id parameter with the START command, the Id generated by the FIERA CCD DCS is just an Id used in the communication between the FIERA CCD sw and external sw (e.g. INS OS). In particular:

3. The OS expoId follows the directions of [3] and is generated by OS. It is used to communicate between OS and the software on top of it (GUI panel, Sequencer etc.).
4. The CCD expoId is a different variable and is generated by CCD at `START` time. It is used only for the internal communication between CCD sw and higher level sw (e.g. INS OS or TCS AG sw).

   In particular the following points should be considered:

   a. Command `SETUP`. For exposures not started yet (command `START` not sent yet), always use `fcdEXP_NEXT` (-1) as expoId parameter.

b. Command `START`. No Id has to be passed as parameter. However, the reply contains an Id.
c. Other commands related to a running exposure (`ABORT` etc.). They need an Id as a parameter. If only one exposure is still running (normal cases) one can use always `fcdEXP_LAST` (0) as Id. If more exposures are running in different phases (quite unusual), then the value returned by the FIERA CCD sw to the `START` command has to be passed as Id.

2.3.2 Exposure types

The FIERA CCD DCS distinguishes among different types of exposure (see [5]).

1. **Dark**. It has the following characteristics:
   a. It consists of one single integration
   b. The CCD is normally wiped before starting a new exposure (determined by read-out mode).
   c. During the integration the shutter is kept closed.
   d. After the integration, the CCD is read out. The read-out mode is determined by setup parameters.
2. **Normal**. It has the following characteristics:
   a. It consists of one single integration
   b. The CCD is normally wiped before starting a new exposure (determined by read-out mode).
   c. During the integration the shutter, if present, is kept open.
   d. After the integration, the CCD is read out. The read-out mode is determined by setup parameters.
3. **Multi-step**. It has the following characteristics*:*
   a. It consists of many integrations. They may have the same or different duration.
   b. The CCD is normally wiped before starting a new exposure (determined by read-out mode).
   c. During the integration the shutter, if present, is kept open.
   d. After each integration, excluding the last one, the exposure is automatically paused (the shutter, if present, is closed)
   e. During pauses between consecutive integrations, rows may be shifted on chip, according to the value of the setup parameter `DET.READ.SHIFT` (see also `fcdREADSHIFT` in file *fcd.h*).
   f. The next integration is started with a CONT command.
   g. After the integration, the CCD is read out. The read-out mode is determined by setup parameters.
4. **Drift Scanning** (*not supported by the present release*).

It is the responsibility of higher level software (e.g. Observation Software) to translate the various supported exposure types as defined in [4] into one of the types known by FIERA CCD DCS for the setup parameter `DET<i>.EXP.TYPE`. We recommend using variables of the type `fcdEXPTYPE` or the macros associated with the corresponding strings (e.g. `fcdEXP_DARK_STR`), defined in *fcd.h*, for all

operations dealing with the FIERA CCD DCS exposure type.

2.3.3 Exposure status

An attribute in the on-line database (*exposures:exposure<index>.expStatus*, see also section 2.14.4) is dedicated to the storage of the status of a running exposure (or the end status of the last finished exposure if no new exposure has been started yet). It is a bit-field value and the meaning of each bit is (least significant bit is indicated as 1):

1. NOT ACTIVE,
2. PENDING,
3. INTEGRATING,
4. PAUSED,
5. readout ACTIVE,
6. PROCESSING IMAGE DATA,
7. TRANSFERRING IMAGE DATA,
8. COMPLETED SUCCESSFULLY,
9. COMPLETED WITH ERROR,
10. ABORTED,
11. FINITE LOOP OF REPEATED EXPOSURES ACTIVE,
12. INFINITE LOOP OF REPEATED EXPOSURES ACTIVE,
13. WIPING.

Note that, in case of a loop of repeated exposures, the status does NOT change during the whole loop. Applications interested in the status of each single exposure should not use the repeated exposure feature.
The exposure status returned by the FIERA CCD sw in the reply of some commands (e.g. `STATUS` and `WAIT`) is coded in the same way. Additionally, for those exposures already completed, whose final status is not available anymore (e.g. because a new exposure has been started), the returned status is `fcdEXP_DONE` (see *fcd.h*)
See also file *fcd.h* for all macros and associated values (e.g. `fcdEXP_INACTIVE`).

2.4 Operational modes and simulation

A detailed description of the operational modes implemented and their meaning is given in Chapter 8.

The operational mode can be set:

1. Programmatically.
   - Write directly in the Workstation on-line database (see section 2.14.4).
   - Run Workstation script *fcdDcsDbSave.sh* (see section 4.3).
2. Interactively through a GUI panel (see Chapter 8 or manual page of *fcdOpMode* for more):
   > fcdOpMode "cwp=<alias>$CCDNAME" &

The current operational mode can be retrieved through the command *STATUS* (see 2.11).

We recommend using variables of the type `fcdOPMODE`, defined in *fcd.h*, for all operations dealing with the FIERA CCD sw operational mode

Note:Any change to the current operational mode takes effect only after the FIERA CCD sw is brought to *OFF* state (Shutdown, see also 2.5).

2.5 Operational states

The list of operational states implemented in the FIERA CCD sw is given in section 8.7. In Table 1 are shown the commands needed to modify in the operational state of a CCD camera.

Table 1 FIERA CCD DCS state transition commands

From \ To	OFF	LOADED	STANDBY	ONLINE
OFF	---	fcdDcsStart.sh	---	---
LOADED	OFF	---	---	---
STANDBY	OFF	---	STANDBY	ONLINE
ONLINE	OFF	---	STANDBY	ONLINE

See section 2.11 for more information about commands available.
The current operational state can be retrieved through the command `STATUS` (see 2.11).
We recommend using variables of the type `fcdSTATE`, defined in *fcd.h*, for all operations dealing with the FIERA CCD sw operational state.

2.6 Image processing

The FIERA CCD sw provides facilities to perform real-time image processing on the FIERA SLCU while the image data are being readout and before they are transferred to Workstation.
Currently these facilities consist of:

1. Computation of minimum and maximum pixel values in the image (see section 2.14.2 on setup parameter `DET<i>.WIN<i>.MINMAX`). The results are stored in the SLCU on-line database in the same sequential order as below:
   a. ***images:process:window<index>.ipXMin***. X coordinate of pixel with the minimum value.
   b. ***images:process:window<index>.ipYMin***. Y coordinate of pixel with the minimum value.
   c. ***images:process:window<index>.ipMinVal***. Minimum pixel value in the frame.
   d. ***images:process:window<index>.ipXMax***. X coordinate of pixel with maximum value.
   e. ***images:process:window<index>.ipYMax***. Y coordinate of pixel with maximum value.
   f. ***images:process:window<index>.ipMaxVal***. Maximum pixel value in the frame.
   g. ***images:process:window<index>.ipMean***. Average pixel value in the frame.
2. Computation of rms of the pixel values in the image (see section 2.14.2 on setup parameter `DET<i>.WIN<i>.RMS`). The result is stored in the SLCU on-line database in:
   a. ***images:process:window<index>.ipRMS***.

3. Calculation of centroiding over an image. It does not implement any pattern recognition algorithm; it simply first subtracts the background level, then applies a threshold (all pixels below the threshold are considered to be 0), and finally computes the centre of gravity of the resulting image. The way that the background and threshold are determined is discussed in section 2.14.2 (setup parameter `DET.WIN<i>.CENTROID`).
4. Additional processing. An additional function may be invoked after the centroiding algorithm, currently the function `IQE` is implemented which performs a gaussian fit to determine an accurate centroid and full-width half-maximum. see section 2.14.2 (setup parameter `DET.WIN<i>.IPFUNC`).

Steps 1-4 can be individually enabled or disabled. If more than one is enabled they are performed sequentially in the order given above. The results of the centroiding and IQE function are stored in the attributes below:

a. ***images:process:window<index>.ipXCen***. Difference between X coordinate of centre of gravity and reference position.
b. ***images:process:window<index>.ipYCen***. Difference between Y coordinate of centre of gravity and reference position.
c. ***images:process:window<index>.ipCenVal***. Intensify of the pixel closest to centre of gravity.
d. ***images:process:window<index>.ipBackGnd***. Background intensity.
e. ***images:process:window<index>.ipXFWHM***. X Full-width half-maximum
f. ***images:process:window<index>.ipYFWHM***. Y Full-width half-maximum
g. ***images:process:window<index>.ipNumPix***. Number of pixels over the threshold value

If no object is found during the centroid or IQE function, all attributes are set to 0.

2.7 Telemetry Monitoring

The FIERA CCD sw provides facilities to monitor telemetry (temperature and pressure) values from the detector.

Telemetry is configured using the configuration panel *fcdConfigTel,* and can be displayed using the panel *fcdTelemetry*.

The telemetry sub-system is enabled and disabled using the commands `STARTTL` and `STOPTL`. The current telemetry values are available in the database attribute ***telemetry.current***.

2.8 Startup

The FIERA CCD DCS software provides a **cold startup** script. It performs a complete startup, including operations which are needed only once at the beginning of a session, such as loading the on-line database with the configuration values for the camera used. This script has to be run at least once at the beginning of a session.
Type from the Workstation shell (see also manual page at section 4.3):

```
> fcdDcsStart.sh
```

Additionally, the FIERA CCD DCS software provides a **warm startup** script, which only schedules the CCD processes, and assumes that the rest (database, CDTs, scan system) has already been configured.

Type from the Workstation shell (see also manual page at section 4.3):

```
> fcdDcsWarmStart.sh
```

2.8.1 Startup verification

To verify if the FIERA CCD sw startup was successful:

1. The operational state (see section 2.5) should be 2 (*fcdLOADED*). Check from the UNIX shell:
   > dbRead "<alias>$CCDNAME.opState"
   INT32/UINT32 value = 2
2. The following processes must be running in the WS environment (use option *Rtap Perf. Monitor* from *ccsei* to check)
   a. fcdconCI_*myccd*
   b. fcditWs_*myccd*
3. The following processes must be running in the SLCU environment (use option *Rtap Perf. Monitor* from *ccsei* to check):
   a. fcdcon_*myccd*
   b. fcdit_*myccd*
   c. fcdexp
   d. fcdint_*myccd*
   e. fcdpServer

2.9 Shutdown

The FIERA CCD DCS software provides a shutdown script.
Type from the Workstation shell (see also manual page at section 4.3):

```
> fcdDcsStop.sh
```

2.9.1 Powering Down the FIERA SPARC

If you intend to power down the FIERA SPARC, it is essential that the SPARC is " halted"first.
Type from the Workstation Shell:

```
> fcdlDcsSlcuHalt.sh
```

After the connection has been closed, wait 1 minute, it is then safe to power down the SPARC. The SPARC will boot automatically when it is powered up again.

2.9.2 Rebooting the FIERA SPARC

If you intend to reboot the FIERA SPARC, type from the Workstation Shell:

```
> fcdlDcsSlcuReboot.sh
```

2.10 Include files

Applications interfacing to the FIERA CCD DCS software have to include the file *fcd.h* (see also 4.5).

2.11 Command Interface

The only FIERA CCD process accessible via CCS commands from external software is `fcdconCI_$CCDNAME`, running on the Workstation (e.g. if `CCDNAME` is set to `myccd`, the process name is `fcdconCI_myccd`)**.**
The command interface between the FIERA CCD software and external software is represented by the Command Definition Table of that process.
Here a simple list of all the commands foreseen is given. For more details, see Command Definition Table at section 4.2.1

1. **Abort.** Abort last exposure started or a specified one.
2. **Cont.** Continue an exposure paused by the user or automatically if multi-step.
3. **Dump.** Dump last image read or the specified one from memory. If no image available, read the chip(s) and then transfer to WS (*not yet implemented*).
4. **End.** Stop current integration and read chip immediately.
5. Exit. Same as *Off.*
6. **Kill.** Kill the FIERA CCD main task.
7. Off. Bring the system to OFF state. Terminate the entire CCD software.
8. Online. Bring the system to ON-LINE state.
9. **Pause.** Pause current integration.
10. **Setup.** Define or check setup values for next exposure or change values for the specified running exposure. See also sections 2.14.2 and 4.7.1.
11. **Standby.** The whole FIERA CCD system is brought to STAND-BY state.
12. **Start.** Start new exposure (possibly repeated). Setup parameters can be specified as well. One single reply is returned as soon as the exposure has started.
13. **StartTelemetry.** Start monitoring of telemetry values.
14. **StartWipe.** Start periodical wiping of the chip(s). Setup parameters can be specified as well.
15. **Status.** Check and update current status. Return information in the reply.
16. **Stop**. Stop in an ordered way an ongoing loop of repeated exposures.
17. StopTelemetry. Stop monitoring of telemetry values.
18. **StopWipe.** Stop periodical wiping of the chip(s).
19. Version. Return the current version of the FIERA CCD sw.
20. Wait. Wait for the specified exposure to complete.

External applications are recommended to use the macros for commands and parameters as specified in *fcd.h* (e.g. command `fcdCMD_SETUP` parameter `fcdEXP_TYPE`)

2.12 C libraries

The library *libfcd* provides routines for external software:

1. ***fcdGetConf***. Get the current camera configuration (see also types `fcdCAMERA` and `fcdCONFIG` in *fcd.h*). See manual page in section 4.1
2. ***fcdGetCIName***. Get name of the CCD process to send commands to (see also section 2.11). See manual page in section 4.1
3. ***fcdGetIndexFromId*.** Get index of database point for an exposure from its Id. See manual page in section 4.1

4. **fcdCheckSetup**. Check a complete setup and return computed values (see types `fcdSETUP` and `fcdSETUPRES` in *fcd.h*). See manual page in section 4.1

2.13 Graphical User Interface CCD classes

Applications using the FIERA CCD sw are provided with a set of classes, produced with the CCS panel editor, to be instantiated in bigger application panels.
The classes provided are available in the public library ***libfcdGuiPublic.tcl*** (which has to be registered to any panel using it, see [12]):

1. ***fcdExpStatus_uifClass***. It displays the status of an exposure. See also Fig.2 and manual page at section 4.4

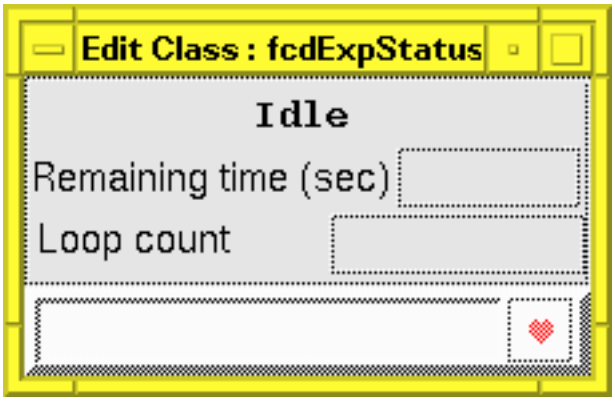


Fig.1 Exposure status GUI class

2. **fcdExpSetup_uifClass.** It displays the main setup parameters of the most recently started exposure. See also Fig.2 and manual page at section 4.4

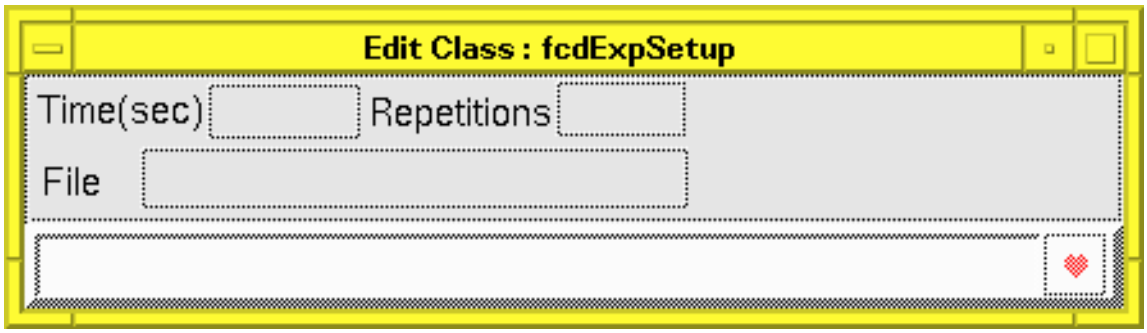


Fig.2 Exposure setup GUI class

3. **fcdReadoutSetup_uifClass.** It displays the readout setup parameters of the most recently started exposure. See also Fig.2 and manual page at section 4.4

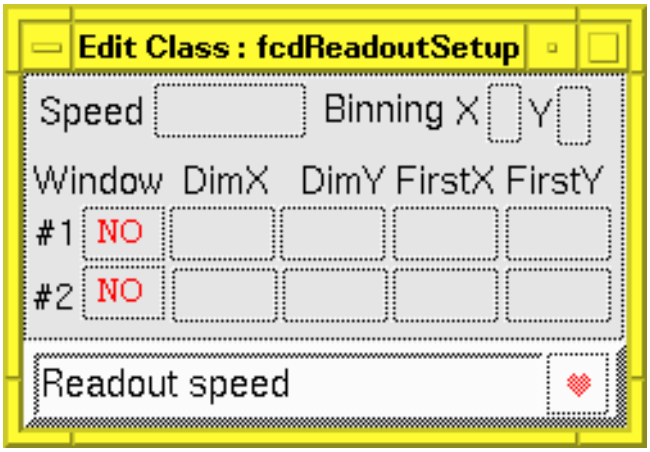


Fig.3 readout setup GUI class

4. *fcdIpStatus_uifClass*. It displays the status of real-time image processing on SLCU. See also Fig.4 and manual page at section 4.4.

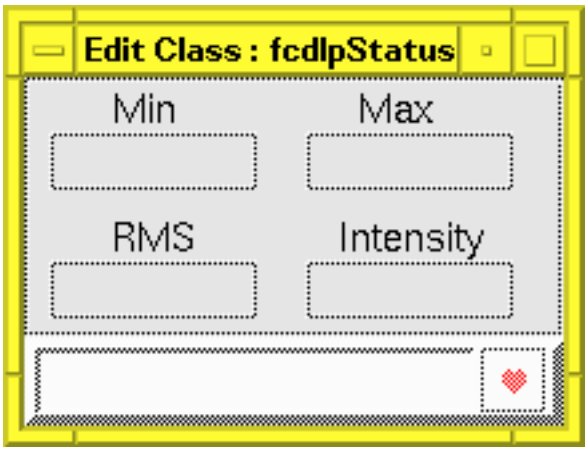


Fig.4 Real-time image processing status GUI class

2.14 Data

2.14.1 Data Interface Dictionary

The FIERA CCD sw needs two Data Interface Dictionary files to be able to operate with setup keywords and files and to create proper headers in FITS files:

1. The first dictionary is the generic CCD Dictionary (see [18]). The name of this dictionary is specified through the environment variable `CCDDID` (see [18] and section 5.6.1). The default generic CCD dictionary (if `CCDDID` is not defined) is `CCDDCS` (full name `ESO-VLT-DIC.CCDDCS`).
2. The second dictionary is the FIERA CCD Dictionary. The name of this dictionary is specified through the environment variable `FCDDID` (see section 5.6.1). The default FIERA CCD dictionary (if `FCDDID` is not defined) is `FCDDCS` (full name `ESO-VLT-DIC.FCDDCS`).

REMARK: These files still need approval from the ESO Data Interface Control Board. The contents is just a proposal and might be affected by substantial changes.

2.14.2 Setup parameters and files

Among all kinds of setup files defined in [3], the FIERA CCD sw handles only detector setup files. They are normally created and modified through GUI panels and can be used as input for exposure setup definition (setup file name is passed as parameter to the SETUP command as well as single setup keywords).

Currently not all setup keywords as defined in the CCD DCS Data Dictionaries (see 2.14.1) are implemented in the FIERA CCD DCS software. The keywords implemented are listed in the setup file shown in section 4.7.1 (at this stage it represents a complete setup file for FIERA CCD DCS). Information about the parameters for the SETUP command are also given in the CDT of fcdconCI (section 4.2.1) and in the include file *fcd.h* (section 2.10)

Some remarks on the implemented setup parameters:

1. The parameter `DET<i>.EXP.TYPE` can have the following string values (see section 2.3.2 and *fcd.h*):
   a. *fcdEXP_NORMAL_STR*
   b. *fcdEXP_DARK_STR*
   c. *fcdEXP_MULTI_STR*
   d. `fcdEXP_DRIFT_STR` (*Not implemented yet)*
2. The parameter `DET<i>.EXP.NREP` indicates how many times the defined exposure must be executed.

   The special value `fcdREPEAT_FOREVER` (see *fcd.h*) defines an infinite loop of repeated exposures.

   Applications interested in knowing the intermediate status of an exposure must set `DET<i>.EXP.NREP` to 1 (no repetition).
3. The parameter `DET<i>.EXP.TIMEREP` defines the period (in secs) between consecutive exposures for repeated exposures. A value of 0 means no delay between exposure (next started as soon previous one finishes).

   It is ignored in case of single exposures (`DET<i>.EXP.NREP` set to 1).
4. The parameter `DET<i>.FRAM.TYPE` indicates the type of frame resulting from the next exposure. The SLCU CCD sw keeps in memory at any time the last acquired image until it is overwritten by a new exposure. The image can be retrieved at any time by the DUMP command (e.g., in case of image transmission failures or problems on the instrument workstation).
5. The parameter `DET<i>.WIN1.UIT<i>` specifies the integration time in seconds of each integration step. If `DET.EXP.TYPE` is not `fcdEXP_MULTI_STR`, only `UIT1` is used.
6. The parameter `DET<i>.WIN1.NDIT` specifies the number of sub-integrations to be performed. The maximum value permitted by the CCD S/W is 10 (`fcdMAXINTEGR` in fcdDbPublic.h). This value is used only if `DET.EXP.TYPE` is set to `fcdEXP_MULTI_STR`.
7. The parameter `DET<i>.WIN1.ASUIT1` is ignored if `DET<i>.EXP.TYPE` is not set to `fcdEXP_MULTI_STR`. It is of logical type (`T/F`). If set to `T`, all sun-integrations are given the same time as specified for `DET<i>.WIN1.UIT1`.

8. The parameter *DET<i>.READ.SHIFT<i>* specifies the amount of rows to be shifted after a sub-integration. Lines are shifted towards the serial register specified by the *DET<i>.READ.CLKIND*.

   It is ignored if *DET<i>.EXP.TYPE* is not *fcdEXP_MULTI_STR.*

9. The parameter *DET<i>.READ.SHIFTYP* determines which type of on-chip rows shift has to be applied between sub-integrations. It is ignored if *DET<i>.EXP.TYPE* is not *fcdEXP_MULTI_STR*. Possible values are (see *fcd.h*):

   a. *fcdLINE_SHIFT_IDEM_STR.* Shift always in the same direction the amount of rows specified in *DET<i>.READ.SHIFT1*.

10. The parameter *DET<i>.READ.CLKIND* is an integer number (range 0-19). It determines which readout mode has to be used. This choice determines automatically the readout speed and the number and location of on-chip outputs used (see also panel *fcdConfig* to find the correspondence between this index and the characteristics of the associated readout). This choice does NOT determine the binning to be applied to the readout, which is set by *DET<i>.WIN1.BINX* and *DET<i>.WIN1.BINY* (see below).

11. The parameters *DET<i>.WIN1.BINX* and *DET<i>.WIN1.BINY* define the binning to be applied to the readout.

    In the current release of the FIERA CCD software, the same binning factor (set by *DET<i>.WIN1.BINX* and *DET<i>.WIN1.BINY)* applies to all defined windows.

12. The parameter *DET<i>.FRAM.FITSMTD* indicates if and how an image is to be saved on disk in FITS format. Currently only *fcdDISK_NONE* and *fcdDISK_UNCOMPRESS* are implemented (see *fcd.h*).

13. The parameter *DET<i>.FRAM.FITSUNC* specifies the name of the file. It must have suffix ***.fits.*** The maximum allowed length is *fcdMAXLENFILE* (see *fcd.h).*

    This parameter is ignored if *DET<i>.FRAM.FITSMTD* is set to *fcdDISK_NONE*.

14. The parameters *DET<i>.DISPLAY* specifies the frame Id for the VLT Real-Time display facility. The value *fcdNO_DISPLAY* (see *fcd.h*) indicates that images do not need to be displayed.

15. The parameter *DET<i>.FRAM.SAMPLE* indicates how frequently images have to be transferred to the Workstation during a loop of repeated exposures (e.g., for Adaptive Optics systems). A value of 1 means that all images have to be transferred. A value *N* higher than 1 means that only one image out of *N* has to be transferred. Useful to reduce network load if no high refresh rate is needed on Workstation.

    It is ignored for single not repeated exposures (*DET<i>.EXP.NREP* set to 1) or repeated exposures where images are not needed on Workstation (*DET<i>.FRAM.FITSMTD* set to *fcdDISK_NONE* and *DET<i>.FRAM.DISPLAY* set to *fcdNO_DISPLAY*).

16. The parameter *DET<i>.WIN<i>.ST* is of logical type (*T/F*) and indicates if a windowed readout is wanted. Currently the system supports a maximum of two windows (*DET<i>.WIN1.ST* and *DET<i>.WIN2.ST*). There are however limitations to be taken into account:

    a. in the current release of the FIERA CCD software, windowing is supported ONLY if the whole chip readout is performed through one single output, i.e. for some values, varying from system to system, of *DET<i>.READ.CLKIND*, windowing may not be supported.

17. The parameters *DET<i>.WIN<i>.STRX, DET<i>.WIN<i>.STRY, DET<i>.WIN<i>.NX* and *DET<i>.WIN<i>.NY* define the location (referred to the *first pixel* of the chip, see section 1.6) and size of a window.

    They are ignored if *DET<i>.WIN1.ST* is F.

18. The parameter *DET<i>.WIN<i>.MINMAX* is of logical type (*T/F*) and indicates if statistics (minimum/maximum, average) has to be computed on the data for the specified window.
19. The parameter *DET<i>.WIN<i>.RMS* indicates if and where the rms has to be computed on the data for the specified window (*0* for no rms, *1* for rms on the whole window, *2* for rms on the central quarter of the window).
20. The parameter *DET<i>.WIN<i>.CENTROID* indicates the kind of centroiding algorithm wanted for the specified window. The following values are supported (see *fcd.h*):
    a. *fcdCEN_NONE_STR*: no centroid is performed,
    b. *fcdCEN_STANDARD_STR*: standard centroiding calculation (superseded),
    c. *fcdCEN_THRESHOLD_STR:* threshold based algorithm.
    background level is defined by the value of parameter *DET<i>.WIN<i>.BACKGND.*
    threshold level is defined by the value of parameter *DET<i>.WIN<i>.THRMIN.*
21. The parameter *DET<i>.WIN<i>.BACKGND* defines the background level to be used in the centroiding algorithm when parameter *DET<i>.WIN<i>.CENTROID* has value *fcdCEN_THRESHOLD_STR*. It can have any positive value or the following negative values with special meaning (see *fcd.h*):
    a. *fcdBCKGND_FLUX_WINDOW_SELF*: background level is set to the average value over the related window.
    b. *fcdBCKGND_FLUX_WINDOW_PREV*: background level is set to the average value over window 1 (meaningful only if set for *DET<i>.WIN2.BACKGND* and two windows are going to be read-out).
22. The parameter *DET<i>.WIN<i>.THRMIN* defines the threshold level to be used in the centroiding algorithm when parameter *DET<i>.WIN<i>.CENTROID* has value *fcdCEN_THRESHOLD_STR*. It can have any positive value or the following negative values with special meaning (see *fcd.h*):
    a. *0>N>-10*: threshold level is set to N *σ (σ = standard deviation over the related window),
    b. *-10>N>-20*: threshold level is set to N *σ_1 (σ_1 = standard deviation over the window 1). Meaningful only if set for *DET<i>.WIN2.THRMIN* and two windows are going to be read-out.
23. The parameter *DET<i>.WIN<i>.IPFUNC* indicates the name of the additional function the CCD SLCU sw has to call as last step of the image processing for the specified window. If set to *fcdIP_NO_USER_FUN*, no user function is called. See also section 2.6.
    Currently only the function " Image Quality Estimate", IQE is implemented.
24. The parameters *DET<i>.WIN<i>.IPLLX*, *DET<i>.WIN<i>.IPLLY*, *DET<i>.WIN<i>.IPURX* and *DET<i>.WIN<i>.IPURY* define the offsets with respect to the defined window (see *DET<i>.WIN<i>.STRX* etc.), identifying the sub-window on which the image processing, such as centroiding, has to be performed. *IPLLX* and *IPLLY* define respectively the horizontal and vertical offset with from the lower left corner, *IPURX* and *IPURY* the same with respect to the upper right corner.
25. The parameters *DET<i>.WIN<i>.REFX* and *DET<i>.WIN<i>.REFY* define the reference point (in absolute coordinates) for the computation of the error vector in the centroiding algorithm.
    They are ignored if *DET<i>.WIN<i>.CENTROID* is *fcdCEN_NONE_STR*.

Applications handling CCD setup parameters are requested to use the macros defined in *fcd.h* (e.g. *fcdEXP_TYPE*).

2.14.3 Image data

Image data are provided by the FIERA CCD DCS software in two ways:

- Raw-data for Real-time display.
- FITS files

The orientation of the image is independent from the readout mode used (see request from FORS VLT-LET-VIC-13110-0030): independently from the outputs involved in the readout, the image is displayed and/or stored on file always with the same orientation.
In the current release of the FIERA CCD software, this means that for some readout mode the whole chip must be readout before displaying and writing to disk the very first pixel; in other words, the overall performance of the system depends heavily on the readout mode used, as in some cases image readout and image storage must be performed sequentially, in other cases they can be done in parallel.
The FIERA CCD sw does not *know* the orientation of the camera with respect to the sky. It *thinks* exclusively in terms of physical coordinates on the chip.
Location (1,1) corresponds to the first pixel value stored in a FITS file.

2.14.3.1 Proposed "image extension per port" format

A new FITS format has been proposed, allowing the usage of " imageextensions" per port. This format has not been officially defined yet (details of this format are still under discussion with the DMD), therefore its use is recommended only for test purposes.
This format orders data by output i.e. each output used for readout corresponds to an extension. A primary header sits on the top of the file. In order to use this FITS format, it is necessary, using the configuration panel, to set the relevant option (see 8.1). It is also required to define the `FCD_IT_HOST` environment variable (see 5.6.1).

2.14.3.2 Raw-data for real-time display

the FIERA CCD sw provides **raw data** for the VLT sw **real-time display** utility whenever the setup parameter `DET<i>.DISPLAY` is set to a value higher than `fcdNO_DISPLAY`. The setting of this parameter determines the frame where the image will be displayed (currently the utility *rtd d*efines frame Id 0 for the big frame and 4 for the rapid frame, see [16]). The FIERA CCD sw supports up to 10 different frames Id during the same session (10 different images displayed on different frames).
The mechanism to deliver raw data is the same as defined in [16].
Raw-data are written in shared memory as they come out from the Detector Electronics, namely with full resolution (16-bits unsigned integer). No reduction (e.g. to 8-bits) is done by the FIERA CCD sw.

In addition to the display of the raw-data, the FIERA CCD sw supports also the **display of World Coordinates** through *rtd*. One point in the CCD branch of the OLDB is dedicated to this feature. It contains two different kind of attributes:

1. Attributes having static values, i.e. they do not change from one exposure to another. They can be set once and forever through the script *fcdDcsWcs.sh* (see section 4.3).
2. Attributes having values, which may change from one exposure to another. They are part of the public part of the FIERA CCD database (see also section 2.14.4). It is responsibility of applications using CCDs to enter in these attributes the correct values at the appropriate time.

2.14.3.3 FITS files

Images, as result of exposures, are written on WS disk in FITS format (see [3]), whenever the setup parameter `DET<i>.FRAM.FITSMTD` is other than `fcdDISK_NONE` (see `fcdDISKSAVE` in *fcd.h*).

Currently the only format supported for pixels values is 16-bits signed.
Independently from the readout mode used, the complete physical image is stored in one single FITS file. For multiple windows (currently a maximum of two *not overlapping windows* - see definition in section 1.6 - is supported), they are also stored in one FITS file with IMAGE extension (see [4]).
NOTE: The file format for multiple port readout is under discussion.
Apart from the image raw data, the FIERA CCD sw is also responsible for providing keywords for the FITS header. Depending on their type, keywords are treated in two different ways.

- **Standard keywords**. Some basic keywords, needed by any image analysis system to read the FITS file, are written at the beginning of the file directly by the FIERA CCD sw, such that, whatever happens to higher level software (e.g. OS), a readable FITS file is saved, although with basic information only. The number of 80-characters lines reserved by the FIERA CCD sw at the beginning of the file is defined by the macro `fcdDCSHEADLINES` in *fcd.h*. See also the example given in section 4.8.1
- **Hierarchical keywords**. They are not strictly needed to interpret the pixel values and normally do not appear at the beginning of the FITS header. Since the FIERA CCD sw cannot know at which position in the FITS header they must be written, they are written into a separate file with the same name as the image file and extension defined by the macro `fcdFITSHIERSUFF` in *fcd.h*; it is responsibility of the higher level software (e.g. OS for an instrument) to read this file and merge the information contained with the other information collected from the various equipment (e.g. instrument, telescope).

FIERA CCD DCS first writes in these files, then sets the exposure status to `fcdEXP_COMPLETED`.

The responsibility to build image FITS files is shared among DCS and OS. The following rules are set by the FIERA CCD DCS software:

1. OS sets in the FIERA CCD database, before the FIERA CCD DCS software is started, the name of the directory where image files must be saved (see 2.14.4 and *fcdConfig* in 8.1 and 9.1.4 for an alternative interactive way).
2. OS creates the image file(s) and reserves enough space for the complete FITS header.
3. OS passes to FIERA CCD DCS the name of the file as setup parameter before an exposure is started (see `fcdCMD_SETUP` and `fcdFILE_UNC` in *fcd.h*).
4. When readout is started, FIERA CCD DCS looks for the file specified by OS (see point 3.) and opens it for writing.

   Note 1: Normally the file should be there and have the size for the complete FITS header (see point 2.). If it is not there, FIERA CCD DCS creates it. If it is there, but not accessible (e.g. write protected), FIERA CCD DCS tries to create a new one with the same name, followed by a sequential integer index, starting from 1, incrementing the index until it succeeds.

   Note 2: In case the number of FITS file to be produced is more than one (e.g. setup parameter exposure repetition factor, see `fcdREPEAT_DEF` in *fcd.h*, is set to 3), FIERA CCD DCS assumes that all files will have the same name, followed by a sequential integer index, starting from 0. Example: if

   `fcdREPEAT_DEF` is set to 3,

   `fcdFILE_UNC` is set to *myImage.fits*,

   FIERA CCD DCS will look for files *myImage.fits* (first exposure), *myImage.1.fits* (second exposure) and *myImage.2.fits* (third exposure)

   See also [4] for file naming rules.

5. FIERA CCD DCS writes the basic standard keywords at the beginning of the file and then moves up to the end of the file (space reserved by OS for the rest of the header).
6. FIERA CCD DCS writes the complete image.

   If multiple window readout has been performed, FIERA CCD DCS writes the first extension header after the first frame data, reserves as much space as in the main header and then writes the second frame data.
7. FIERA CCD DCS writes the hierarchical keywords in a separate file, as described above.
8. FIERA CCD DCS sets the exposure status to `fcdEXP_COMPLETED`.
9. OS retrieves from the FIERA CCD database (see section 2.14.4) the name of the file containing the image. Normally it is the same as the setup values passed by OS, but FIERA CCD DCS may have been obliged to change it (see point 4. above).
10. OS collects hierarchical information from all sub-systems, moves `fcdDCSHEADLINES` from the beginning of the file and writes them. If multiple window readout has been performed, the same operation has to be done for all extension headers.

    OS knows how much space it has reserved for the header and must ensure that the image data section is not overwritten with header information.

Note: tools and functions for handling image FITS files are provided by the INS common sw (*slx*, see [14], and *oslx*, see [15]).

2.14.4 Public part of the FIERA CCD database

Some attributes of the CCD branch of the on-line database are made public for direct read/write operations from external software. Attributes which are accessed indirectly from external software through commands, functions or panels provided by the FIERA CCD sw (e.g. all setup attributes set with the command SETUP, configuration attributes read with the function *fcdGetConf* and set with the panel *fcdConfig*) are not considered public (no direct call to *dbRead/dbWrite* CCS functions).

When accessing FIERA CCD database attributes with direct CCS db calls, **applications are requested to use the macros defined in *fcdDbPublic.h*** (see 4.5): in this way, any change in name or location of the attribute only requires a new compilation.

All database paths below are meant to be relative to the root point for the FIERA CCD database branch. The attributes marked with **(*)** require that the Scan System is working between CCD SLCU and WS.

1. **Read/Write attributes**
   a. ***.opMode***. (dbINT32) Camera operational mode (`fcdDB_CON_OPMODE`).
   b. ***images.imageDirectory*** (dbBYTES128) Directory where images are stored (`fcdDB_CON_IMGPATH`).
   c. ***wcs.ra*** (dbDOUBLE) Centre right ascension in degrees for World Coordinates display (`fcdDB_WCS_RA`).
   d. ***wcs.dec*** (dbDOUBLE) Centre declination in degrees for World Coordinates display (`fcdDB_WCS_DEC`).
   e. ***detector.chipInstances(<ccdId>:<ccdId>,9:9)*** Value for the CRPIX1 FITS keyword. If this attribute is set to fcdNO_DBCRPIX (99999), the value for CRPIX1 will be computed by FIERA. <ccdId> is the index of the CCD chip (starting from 0).

f. ***detector.chipInstances(<ccdId>:<ccdId>,10:10)*** Value for the CRPIX2 FITS keyword. If this attribute is set to fcdNO_DBCRPIX (99999), the value for CRPIX1 will be computed by FIERA. <ccdId> is the index of the CCD chip (starting from 0).

2. Read only attributes

a. ***.opState*** **(*)** (dbINT32) Camera operational state (on Workstation only). (`fcdDB_STA_SYSTEM`).

b. ***.failureWs*** (dbINT32) Workstation status (on Workstation only) (`fcdDB_STA_FAILURE_WS`).

c. ***.failureSlcu (*)*** (dbINT32) SLCU status (`fcdDB_STA_FAILURE_LCU`).

d. ***exposures:exposure_1.expStatus (*)*** (dbINT32) Status of exposure. Same for ***exposures:exposure_2.expStatus*** (`fcdDB_EXP_POINT_<index>.fcdDB_STA_EXPOSURE`).

e. ***exposures:exposure_1.expId*** (dbINT32) ID of exposure. Same for ***exposures:exposure_2.expId*** (`fcdDB_EXP_POINT_<index>.fcdDB_STA_EXPID`).

f. ***exposures:exposure_1:transfer.fileNameUnComp*** (dbBYTES32) Name of FITS file where uncompressed image data are written (on Workstation only). Same for ***exposures:exposure_2:transfer.fileNameUnComp*** *(`fcdDB_EXP_POINT_<index>:fcdDB_EXP_FILEUNC`).*

g. ***exposures:exposure_next:readout.modeIndex*** (dbINT32) Index of the mode chosen for readout (`fcdDB_SET_MODEINDEX`)

h. ***exposures:exposure_next:readout:window_%d.enabled*** (dbLOGICAL) Readout window enable (`fcdDB_SET_WIN<index>_ENAB`). Same for ***exposures:exposure_1:readout:window_%d.enabled*** (`fcdDB_SET_WIN<index>_ENAB_1`).

i. ***exposures:exposure_next:readout:window_%d.xBinning*** (dbINT32) Horizontal binning factor for readout window (*fcdDB_SET_WIN<index>_XBIN*). Same for ***exposures:exposure_1:readout:window_%d.xBinning*** (`fcdDB_SET_WIN<index>_XBIN_1`)

j. ***exposures:exposure_next:readout:window_%d.yBinning*** (dbINT32) Vertical binning factor for readout window (`fcdDB_SET_WIN<index>_YBIN`). Same for ***exposures:exposure_1:readout:window_%d.yBinning*** (`fcdDB_SET_WIN<index>_YBIN_1`)

k. ***exposures:exposure_next.timeDef*** (dbDOUBLE) Exposure time (`fcdDB_SET_TIMEDEF`). Same for ***exposures:exposure_1.timeDef*** (`fcdDB_SET_TIMEDEF_1`)

l. ***exposures:exposure_next.timeRepeat*** (dbDOUBLE) Exposure repeat time (`fcdDB_SET_TIMEREP`). Same for ***exposures:exposure_1.timeRepeat*** (`fcdDB_SET_TIMEREP_1`)

m. ***exposures:exposure_next.expType*** (dbINT32). Exposure type (`fcdDB_SET_EXPTYPE`). Same for ***exposures:exposure_1.expType*** (`fcdDB_SET_EXPTYPE_1`)

n. ***exposures:exposure_next.expRepeatDef*** (dbINT32) Number of repetitions (`fcdDB_SET_EXPREPS`). Same for ***exposures:exposure_1.expRepeatDef*** (`fcdDB_SET_EXPREPS_1`)

o. ***exposures:exposure_<index>:timeRem (*)*** (dbDouble) Exposure remaining time (`fcdDB_EXP_POINT_<index>:fcdDB_STA_TIMEREM`)

p. ***detector.chipInstances*** (Table of dbBYTES16, dbBYTES16, dbINT32, dbINT32, dbLOGICAL, dbLOGICAL, dbDOUBLE, dbDOUBLE) Table of chip Instances (`fcdDB_CHIP_INST`)

q. ***detector.description*** (dbBYTES16) Detector description (`fcdDB_DET_DESCR`)

r. ***detector.xPixels*** (dbINT32) Number of columns on chip (`fcdDB_CON_XPIX`)

s. detector.yPixels (dbINT32) Number of rows on chip (`fcdDB_CON_YPIX`)

t. ***observations.setupDirectory*** (dbBYTES128) Setup files directory (`fcdDB_OBS_STP_DIR`)

u. ***readout:mode_<index>.description*** (dbBYTES64) Readout mode description (`fcdDB_POINT_RDT_MODE_<index>.fcdDB_CON_MODEDES`)

v. ***readout:mode_<index>.elAdu*** (Vector of dbDOUBLE) Electrons per Adu per output (`fcdDB_POINT_RDT_MODE_<index>.fcdDB_CON_ELADU`)

w. ***readout:mode_<index>.ron*** (Vector of dbDOUBLE) readout nois per output (`fcdDB_POINT_RDT_MODE_<index>.fcdDB_CON_RON`)

x. ***readout:mode_<index>.speed*** (dbBYTES16) Description of readout speed (`fcdDB_POINT_RDT_MODE_<index>.fcdDB_CON_SPEED`)

y. ***images:process:window_i.ipMinVal (*)*** (dbINT32) min pixel value. (`fcdDB_IP_MINVAL_FMT`).

z. ***images:process:window_i.ipMaxVal (*)*** (dbINT32) max pixel value. (`fcdDB_IP_MAXVAL_FMT`).

aa. ***images:process:window_i.ipRMS (*)*** (dbDOUBLE) Rms calculation. (`fcdDB_IP_RMS_FMT`).

ab. ***images:process:window_i.ipXCen (*)*** (dbDOUBLE) Error vector (x-component) (`fcdDB_IP_XCEN_FMT`).

ac. ***images:process:window_i.ipYCen (*)*** (dbDOUBLE) Error vector (y-component) (`fcdDB_IP_YCEN_FMT`).

ad. ***images:process:window_i.ipCenVal (*)*** (dbDOUBLE) Centroid value (`fcdDB_IP_CENVAL_FMT`).

ae. ***images:process:window_i.ipXFWHM (*)*** (dbDOUBLE) Full-width half maximum (x-component) (`fcdDB_IP_X_FWHM_FMT`).

af. ***images:process:window_i.ipYFWHM (*)*** (dbDOUBLE) Full-width half maximum (y-component) (`fcdDB_IP_Y_FWHM_FMT`).

ag. ***images:process:window_i.ipNumPix (*)*** (dbDOUBLE) Number of pixels above threshold level (`fcdDB_IP_NUMPIX_FMT`).

ah. ***images:process:window_i.ipBackGnd (*)*** (dbDOUBLE) Background value (`fcdDB_IP_BACKGND_FMT`).

ai. ***telemetry.current (*)*** (vector of dbDOUBLE) Current telemetry values (`fcdDB_TM_CURRENT`)

aj. ***telemetry.opState (*)*** (dbINT32) Current state of telemetry monitoring (`fcdDB_TM_OPSTATE`)

3. Read only attributes for GUI panels.

As already mentioned in section 2.13, the FIERA CCD sw provides applications with CCD GUI classes to be incorporated in their own panels. We **strongly recommend** using these classes in application panels. For all cases where this is not possible, the following attributes are made public for usage within GUI panels;.

a. ***images:transfer.percent*** (dbINT32) Percentage of image transferred to WS (`fcdDB_IT_STA_PERC`).

b. ***images:transfer.last*** (dbINT32) Last line transferred to WS (`fcdDB_IT_STA_LINE`).

c. ***shutter.status (*)*** (dbINT32) Shutter status (`fcdDB_IT_STA_SHTSTATUS`).

2.15 Alarm System

As part of the telemetry monitoring, if a telemetry value goes out of limits (defined using the Telemetry Configuration Panel, see Fig.7 of Chapter 8) an alarm is raised with the alarm system. The alarm generated is `tmOutOfLimit`. Alarms can be monitored using the standard VLT utility *alrmDisplay*.

2.16 Configuration and operational logs

The FIERA CCD sw logs the main operations it performs, using the VLT CCS Logging System, in FITS format. Examples of logs produced by the FIERA CCD sw are given in section 4.9.
According to the syntax specified in [5], the last part of the log message must contain a source mask, in order to identify the sub-system the log is coming from. As CCD cameras are used in many different sub-systems, this mask cannot be set by the CCD sw itself, but must be defined and set by the sub-system using it. A script, to be executed once when configuring the CCD sw (see section 5.6.5) and called *fcdDcsSetLogMask.sh* (see also section 4.3) is dedicated to this purpose. Example:

> fcdDcsSetLogMask.sh $CCDNAME $RTAPENV “ MSKSU”

2.17 Example of usage of CCD cameras

Assuming that all the environment variables have been properly set (see 5.6.1), in the following example the FIERA software is started and some exposures are performed.

1. Start the FIERA Software from the Instrument Workstation
   >fcdDcsStart.sh $CCDNAME $RTAPENV $CCDLENV $INS_ROOT
2. Put the FIERA Software ONLINE
   >msgSend $RTAPENV fcdconCI_$CCDNAME ONLINE ""
3. Perform periodic wiping
   >msgSend $RTAPENV fcdconCI_$CCDNAME STARTWP ""
4. Perform Telemetry
   >msgSend $RTAPENV fcdconCI_$CCDNAME STARTTL ""
   > dbRead "<alias>${CCDNAME}:telemetry.current"
5. Stop Telemetry
   >msgSend $RTAPENV fcdconCI_$CCDNAME STOPTL ""
6. Prepare the first exposure (send a complete SETUP to the FIERA Software)
   >msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-file fcdSetupComplete.det"
7. Start the exposure
   >msgSend $RTAPENV fcdconCI_$CCDNAME START ""
8. Wait until the exposure has been completed
   >msgSend $RTAPENV fcdconCI_$CCDNAME WAIT "0"
9. Prepare the next exposure (change only the binning factor)

> msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-function DET1.WIN1.BINX 2 DET1.WIN1.BINY 2"

10. Start the exposure

>msgSend $RTAPENV fcdconCI_$CCDNAME START ""

11. Wait until the exposure has been completed

>msgSend $RTAPENV fcdconCI_$CCDNAME WAIT "0"

12. Prepare the next exposure with image statistic

> msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-function DET1.WIN1.MINMAX T"

13. Start the exposure

>msgSend $RTAPENV fcdconCI_$CCDNAME START ""

14. Wait until the exposure has been completed

>msgSend $RTAPENV fcdconCI_$CCDNAME WAIT "0"

15. Read the image processing results

> dbRead "<alias>${CCDNAME}:images:process:window_1.ipMinVal"

> dbRead "<alias>${CCDNAME}:images:process:window_1.ipMaxVal"

> dbRead "<alias>${CCDNAME}:images:process:window_1.ipRMS"

16. Disable image statistic

> msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-function DET1.WIN1.MINMAX F"

17. Prepare the next exposure with image centroid calculation

> msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-function DET1.WIN1.CENTROID threshold"

> msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-function DET1.WIN1.BACKGND 10"

> msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-function DET1.WIN1.THRMIN 100"

18. Start the exposure

>msgSend $RTAPENV fcdconCI_$CCDNAME START ""

19. Wait until the exposure has been completed

>msgSend $RTAPENV fcdconCI_$CCDNAME WAIT "0"

20. Read the image centroiding results

> dbRead "<alias>${CCDNAME}:images:process:window_1.ipXCen"

> dbRead "<alias>${CCDNAME}:images:process:window_1.ipYCen"

21. Disable image centroid calculation

> msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-function DET1.WIN1.CENTROID none"

22. Put the FIERA Software in STANDBY

>msgSend $RTAPENV fcdconCI_$CCDNAME STANDBY ""

23. Exit

>fcdDcsStart.sh $CCDNAME $RTAPENV $CCDLENV $INS_ROOT

3 NOTES FOR ADAPTIVE OPTICS SYSTEMS

This part of the document provides a description of the special features of the programmatic **added** to the interface of the FIERA CCD sw only for the Adaptive Optics systems.
All the contents of the previous chapter 2 are still valid.

3.1 Changing a Readout Mode

A Readout Mode can be executed

1. when the system is not operating (see 3.1.1),
2. when a system is already running a different Readout Mode, i.e, you want to change the Readout Mode used by FIERA (see 3.1.2).

Before running a mode, send a `SETUP` defining the following parameters:

DET.READ.CLKINDindex of the clock pattern to be used

DET.WIN1.BINX binning factor along X

DET.WIN1.BINY binning factor along Y

DET.WIN1.UIT1user defined subintegration time (sec)

DET.FRAM.SAMPLEimage sampling rate to the workstation

To obtain the maximum rate, `DET.WIN1.UIT1` has to be always 0 (in this case the chip will be continuously readout, and the effective "integration time" will be the time needed to readout the chip, since there is no shutter and the light is always reaching the CCD).
To adjust down the frame rate, `DET.WIN1.UIT1` will define the time between two different readouts of the chip (in this case the effective "integration time" will be the time needed to readout the chip PLUS the time defined by `DET.WIN1.UIT1`).

NOTE: RTD can handle a maximum rate of ~10 frames per sec. If this rate is exceeded, RTD fails because of overrun. To avoid this, set `DET.FRAM.SAMPLE` appropriately, in order to send to the SPARC only one each `N` images.

Whenever needed, the STOP command ends the execution of an exposure loop:

```
> msgSend $RTAPENV fcdconCI_$CCDNAME STOP ""
```

3.1.1 execution of a Readout Mode when the system is not operating

About the different Readout Modes and the setups needed before starting the Readout Modes, this is the simplest way to perform them (see also 2.17):

1. Start the FIERA Software from the Instrument Workstation

   ```
   >fcdDcsStart.sh $CCDNAME $RTAPENV $CCDLENV $INS_ROOT
   ```

2. Put the FIERA Software ONLINE

   ```
   >msgSend $RTAPENV fcdconCI_$CCDNAME ONLINE ""
   ```

3. Prepare the first exposure (send a complete SETUP to the FIERA Software)

   ```
   >msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-file fcdSetupComplete.det"
   ```

4. Define which Readout Mode has to be used

>msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-function DET.READ.CLKIND 1 DET.FRAM.SAMPLE 100 DET.WIN1.BINX 1 DET.WIN1.BINY 1 DET.WIN1.UIT1 0.0"

5. START the exposure loop

> msgSend $RTAPENV fcdconCI_$CCDNAME START ""

3.1.2 change the Readout Mode while an exposure loop is running

Assuming that an exposure loop is already running (see 3.1.1):

1. Change the Readout Mode under usage (note the main difference with the example in 3.1.1: here we refer to the **running** exposure loop)

   >msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "**-expoId 0** -function DET.READ.CLKIND 3 DET.FRAM.SAMPLE 30 DET.WIN1.BINX 1 DET.WIN1.BINY 1 DET.WIN1.UIT1 0.0"

2. Change the Readout Mode under usage again

   >msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "**-expoId 0** -function DET.READ.CLKIND 4 DET.FRAM.SAMPLE 60 DET.WIN1.BINX 1 DET.WIN1.BINY 1 DET.WIN1.UIT1 0.0"

3. Change the Readout Mode under usage again

   >msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "**-expoId 0** -function DET.READ.CLKIND 2 DET.FRAM.SAMPLE 100 DET.WIN1.BINX 1 DET.WIN1.BINY 1 DET.WIN1.UIT1 0.0"

3.2 Image Cube

For test purposes, the Adaptive Optics systems can collect image data in ***Image Cubes***, i.e., standard FITS files where more images are stored.

3.2.1 Setup parameter for Image Cubes

In addition to the setup parameters described in 2.14.2, for the Adaptive Optics systems the following parameter has been implemented:

1. The parameter `DET.READ.NFRAM` indicates how many images have to be stored in an *image cube*. The maximum value is 1024. means that all images have to be transferred.

   Note: this setup parameter has to be sent ***during the execution of a loop*** of exposures (see 3.3.1).

3.2.2 How to obtain an Image Cube

During the execution of an exposure loop, a Setup command related to the actual exposure has to be sent to the Exposure Coordination task, with the `DET.READ.NFRAM` parameter defined.

Example:store 20 images in an Image Cube file named `AOCube.fits` (the system has already been set to `ONLINE`, and a complete Setup has already been sent):

1. Define the Image Cube FITS file name

   > msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-function DET.FRAM.FITSUNC AOCube.fits"

2. Start the exposure loop

   >msgSend $RTAPENV fcdconCI_$CCDNAME START ""

3. Acquire an *image cube* of 20 images

   >msgSend $RTAPENV fcdconCI_$CCDNAME SETUP "-expoId 0 -function DET.READ.NFRAM 20"

3.3 Test Mode

To run the Test Mode NO EXPOSURE LOOP MUST BE RUNNING.
To run the Test Mode, use the `SELFTST` command with the followinf syntax:

```
SELFTST -function AOTestFile  <FileName> -repeat <Repetition>
```

where:

`<FileName>` has to be replaced with the *STRING* with the name of the ascii file containing the values which have to be sent by FIERA to the RTC. The file has to be stored in the directory:

$INS_ROOT/SYSTEM/COMMON/CONFIGFILES/$CCDNAME

This file must contain a SINGLE value per line.

`<Repetition>` has to be replaced with the *NUMBER* of times the test has to be repeated.

`Repetition` = 0 means FOREVER

If `Repetition` = 0 the loop can be stopped pushing the "Stop Loop" button in the "General" area of the Fiera Control Panel (see 8.2) or sending the `STOP` command.

3.3.1 Example of Test Mode execution

Assuming that the system is already ONLINE and IDLE (i.e., is NOT running any exposure loop):

1. Perform the Test Mode defined by the file `AOTest.tst` forever

   >msgSend $RTAPENV fcdconCI_$CCDNAME SELFTST "-function AOTestFile AOTest.tst -repeat 0"

2. Stop the execution of the Test Mode

   > msgSend $RTAPENV fcdconCI_$CCDNAME STOP ""

4 REFERENCE

This chapter provides a detailed description of the FIERA CCD software programmatic interface, namely functions, programs (including Command Definition Tables), scripts and include files, together with various template files.

4.1 Functions

Functions provided by the FIERA CCD software are grouped in the library *libfcd.a*, available on Workstation only.

The functions provided by the FIERA CCD software to external software are:

1. **fcdCheckSetup**. It checks the validity of a complete setup and returns also some computed values.
2. **fcdGetCIName**. It returns the name of the FIERA CCD Command Interface process.
3. **fcdGetConf**. It returns all configuration values for the camera used.
4. **fcdGetIndexFromId**. It returns the index of the exposure from its Id.

4.1.1 fcdCheckSetup

NAME

fcdCheckSetup, fcdCheckSetupWindow - Check a setup for a CCD camera

SYNOPSIS

```
#include "fcd.h"

ccsCOMPL_STAT fcdCheckSetup (
                              fcdCONFIG   *config,
                              fcdSETUP    *setup,
                              fcdSETUPRES *results,
                              ccsERROR    *error
                             )

ccsCOMPL_STAT fcdCheckSetupWindow (
                                   fcdCONFIG   *config,
                                   fcdSETUP    *setup,
                                   fcdSETUPRES *results,
                                   ccsERROR    *error
                                  )
```

DESCRIPTION

These functions perform a static check of the defined setup parameters against the configuration. This check does NOT do any dynamic check against the current state of the camera.
The check is stopped after the first error detection and the function returns with FAILURE.

fcdCheckSetup checks a complete setup. It also returns values derived from the setup parameters.

fcdCheckSetupWindow checks only the part of the setup concerning the windowed readout.

<config>	IN	information about camera configuration (see fcdGetConf)
<setup>	IN/OUT	structure containing exposure setup (see ccd.h)
<results>	OUT	structure containing check results (see ccd.h)
<error>	OUT	error structure

RETURN VALUES

SUCCESS if everything OK
FAILURE if error occurs

EXAMPLES

```
#include "fcd.h"
....
fcdCAMERA   camera = {"myccd", ccsLOCAL_ENV};
fcdCONFIG   config;
fcdSETUP    setup;
fcdSETUPRES results;
ccsERROR    error;
....
if (fcdGetConf(&camera, &config, &error) != SUCCESS)
    {
    ... handle failure
    }
```

```
        ....
           fill in values in setup structure
        ....
        if (fcdCheckSetup(&config, &setup, &results, &error) != SUCCESS)
            {
            ... handle failure
            }
```

SEE ALSO

fcdGetConf

- - - - - -

Last change: 01/04/04-08:11

4.1.2 fcdGetCIName

NAME

fcdGetCIName - Get name of Command Interface Process for a CCD camera

SYNOPSIS

```
#include "fcd.h"
void fcdGetCIName(fcdCAMERANAME camera,
                  ccsPROCNAME procName)
```

DESCRIPTION

This function returns the name of the only FIERA process communicating with external sw through the CCS message system to control a specific CCD camera.

<camera>	IN	name of the camera used
<procName>	OUT	name of the Command Interface process for camera used

RETURN VALUES

None

EXAMPLES

```
#include "fcd.h"
....
fcdCAMERA camera = {"myccd", ccsLOCAL_ENV};
ccsPROCESS fcdProcess;
....
fcdGetCIName(camera.name, fcdProcess);
... fcdProcess is now set to "ccdconCI_myccd"
```

- - - - - -

Last change: 01/04/04-08:11

4.1.3 fcdGetConf

NAME

```
fcdGetConf - Get information about a CCD camera configuration
```

SYNOPSIS

```
#include "fcd.h"
ccsCOMPL_STAT fcdGetConf(fcdCAMERA *camera,
                         fcdCONFIG *config,
                         ccsERROR  *error)
```

DESCRIPTION

```
This function returns information concerning the configuration of a
CCD camera, which might be needed by software packages using the
FIERA sw.
<camera>   IN    structure containing the following elements:
                 - name      name of the camera used
                 - envName   name of the CCS environment where
                             FIERA software runs. It can be set to
                             ccsLOCAL_ENV for local env.
<config>   OUT   information about camera configuration
<error>    OUT   error structure
```

RETURN VALUES

```
SUCCESS if everything OK
FAILURE if error occurs
```

EXAMPLES

```
 #include "fcd.h"
 ....
 fcdCAMERA camera = {"myccd", ccsLOCAL_ENV};
 fcdCONFIG config;
 ccsERROR  error;
 ....
 if (fcdGetConf(&camera, &config, &error) != SUCCESS)
     {
     ... handle failure
     }
```

```
- - - - - -
Last change:  01/04/04-08:11
```

4.1.4 fcdGetIndexFromId

NAME

```
fcdGetIndexFromId - extract exposure index from exposure id
```

SYNOPSIS

```
#include "fcd.h"

vltINT32 fcdGetIndexFromId( vltINT32 expId )
```

DESCRIPTION

```
This routine returns the exposure index of the database attribute
exposures:exposure_<expIndex> for the exposure associated to expId

<expId>   IN   exposure id
```

RETURN VALUES

```
Exposure Index (vltINT32)
```

EXAMPLES

```
 #include "fcd.h"
 ....
 vltINT32     expIndex;
 vltINT32     expId;
 ....
 expId = 1231;
 expIndex = fcdGetIndexFromId(expId);
 ... expIndex is 1
```

```
- - - - - -
Last change:  01/04/04-08:11
```

4.2 Programs

The name of all CCD processes within their respective environment is built as *<program name>_$CCDNAME*.

The only program representing the Command Interface to external application software is *fcdconCI*, which runs on Workstation. The corresponding process for the camera *myccd* is therefore *fcdconCI_myccd.*

4.2.1 Command Definition Table for program fcdconCI

The Command Definition Table for the program *fcdconCI_myccd* can be found in `$VLTROOT/CDT/fcdconCI.cdt` (or `$INTROOT/CDT/fcdconCI.cdt`, if `INTROOT` is used).

4.3 Scripts

The scripts available on Workstation are:

1. **fcdInstall.sh**. Installation of files needed by the FIERA CCD sw for the camera used.
2. **fcdDcsStart.sh**. Cold Startup script.
3. **fcdDcsWarmStart.sh**. Warm Startup script.
4. **fcdDcsStop.sh**. Shutdown script.
5. **fcdDcsDbSave.sh**. Save current FIERA CCD configuration.
6. **fcdDcsScan.sh**. Define scan table for FIERA CCD DCS part
7. **fcdosScan.sh**. Define scan table for FIERA CCD standalone part.
8. **fcdDcsWcs.sh**. Set values in the FIERA CCD OLDB for World Coordinates display.
9. **fcdDcsSetLogMask.sh**. Set source mask for FITS logs
10. **fcdDcsSlcuHalt.sh**. Halt the SPARC LCU before shutting it down.
11. **fcdDcsSlcuReboot.sh**. Reboot the SPARC LCU.

4.3.1 fcdInstall.sh

NAME

fcdInstall.sh - Install FIERA files in INS_ROOT

SYNOPSIS

```
fcdInstall.sh <DbFile> [<ins_root_dir>]
```

DESCRIPTION

This shell script installs all files needed to run a FIERA system in the root directory for the instrument it belongs to. Additionally, it stores in the on-line database the values contained in configuration files.

```
<DbFile>       Name of the .dbcfg file in $VLTROOT/config containing DB
               init values.
<ins_root_dir> root directory for the instrument the FIERA belongs to
               Default: $INS_ROOT (env. variable)
```

FILES

$VLTROOT/config/fcd*.dbcfg
Files containing on-line database initialisation values for FIERA systems known.
For scientific CCDs there is one file for each system to be delivered. They normally have to be checked and possibly modified by experts of the specific CCD system. The name is fcdSci<id>.dbcfg whereby <id> is a symbolic name of the system.
Example: "myccd" CCD ---> fcdSciMyccd.dbcfg

ENVIRONMENT

```
CCDNAME  CCD camera name
RTAPENV  FIERA Rtap environment name
INS_ROOT default instrument root directory
INS_USER default to SYSTEM
```

RETURN VALUES

```
0 if SUCCESS
1 if FAILURE
```

EXAMPLES

```
> fcdInstall.sh fcdSciMyccd.dbcfg
Install all what needed for "myccd" scientific CCD
```

SEE ALSO

INS common sw specification

- - - - - -
Last change: 01/04/04-08:11

4.3.2 fcdDcsStart.sh

NAME

fcdDcsStart.sh - startup of FIERA DCS

SYNOPSIS

fcdDcsStart.sh [<camera>] [<WS env.>] [<SLCU env.>] [<INS root>] [kill]

DESCRIPTION

```
This shell script performs a startup of FIERA DCS.
No FIERA stand-alone module is started.
To be run once at startup only!
It performs the following steps:
  1 - Load database configuration values on WS
  2 - Run warm startup (fcdDcsWarmStart.sh)

<camera>    camera name, also root point in DB (default env. var. CCDNAME)
<WS env.>   name of workstation environment (default env. var. RTAPENV)
            If value is 0, no action on WS part of FIERA sw is taken
<SLCU env.> name of SLCU environment (default env. var. CCDLENV)
            If value is 0, no action on SLCU part of FIERA sw is taken
<INS root>  INS_ROOT environment variable
kill        kill all already running processes before starting
```

ENVIRONMENT

```
CCDNAME        default for camera name (e.g. myccd)
RTAPENV        default for WS local environment (e.g. myws)
CCDLENV        default for SLCU environment (e.g myfiera)
INS_ROOT       default root directory for instrument data
```

RETURN VALUES

```
0 if SUCCESS
1 if FAILURE
```

EXAMPLES

```
> fcdDcsStart.sh myccd myws myfiera
      Start the FIERA DCS sw both at WS and SLCU level
      for camera "myccd", WS environment "myws", SLCU environment "myfiera"

> fcdDcsStart.sh myccd myws 0
      Start the FIERA DCS sw at WS level only
      for camera "myccd", WS environment "myws"

> fcdDcsStart.sh myccd myws 0 $INS_ROOT kill
      Kill and restart the FIERA DCS sw at WS level only
      for camera "myccd", WS environment "myws"

> fcdDcsStart.sh myccd 0 myfiera
      Start the FIERA DCS sw at SLCU level only
      for camera "myccd", SLCU environment "myfiera"
```

SEE ALSO

fcdDcsStop.sh fcdDcsWarmStart.sh

```
- - - - - -
Last change:  01/04/04-08:11
```

4.3.3 fcdDcsWarmStart.sh

NAME

```
fcdDcsWarmStart.sh - warm startup of FIERA DCS
```

SYNOPSIS

```
fcdDcsWarmStart.sh [<camera>] [<WS env.>] [<SLCU env.>] [kill]
```

DESCRIPTION

```
This shell script performs a warm startup of FIERA DCS.
No FIERA stand-alone module is started.
To be run once at startup only!
It performs the following steps:
  1 - Terminate all running processes
  2 - Set SLCU sw in appropriate simulation mode
  3 - Schedule processes
  4 - Enable scanning of data from SLCU to WS

<camera>    camera name, also root point in DB (default env. var. CCDNAME)
<WS env.>   name of workstation environment (default env. var. RTAPENV)
            If value is 0, no action on WS part of FIERA sw is taken
<SLCU env.> name of SLCU environment (default env. var. CCDLENV)
            If value is 0, no action on SLCU part of FIERA sw is taken
kill        kill all already running processes before starting
```

ENVIRONMENT

```
CCDNAME  default for camera name (e.g. myccd)
RTAPENV  default for WS local environment (e.g. myws)
CCDLENV  default for SLCU environment (e.g myfiera)
INS_USER default SYSTEM
INS_HOST Host where the INS_ROOT directory is mounted. Default HOST
```

RETURN VALUES

```
0 if SUCCESS
1 if FAILURE
```

EXAMPLES

```
> fcdDcsWarmStart.sh myccd myws myfiera
      Start the FIERA DCS sw both at WS and SLCU level
      for camera "myccd", WS environment "myws", SLCU environment myfiera"

> fcdDcsWarmStart.sh myccd myws 0
      Start the FIERA DCS sw at WS level only
      for camera "myccd", WS environment "myws"

> fcdDcsWarmStart.sh myccd myws 0 kill
      Kill and restart the FIERA DCS sw at WS level only
      for camera "myccd", WS environment "myws"

> fcdDcsWarmStart.sh myccd 0 myfiera
      Start the FIERA DCS sw at SLCU level only
      for camera "myccd", SLCU environment "myfiera"
```

SEE ALSO

```
fcdDcsStart.sh fcdDcsStop.sh
```

```
- - - - - -
Last change:  01/04/04-08:11
```

4.3.4 fcdDcsStop.sh

NAME

```
fcdDcsStop.sh - shut-down FIERA DCS
```

SYNOPSIS

```
fcdDcsStop.sh [<camera>] [<WS env.>] [<SLCU env.>] [kill]
```

DESCRIPTION

```
This shell script performs a shut-down of FIERA DCS.
It does the following steps:
  1 - Verify if the main process is running.
  2 - Try to terminate FIERA processes in a 'soft' way (command EXIT)
  3 - Try to terminate FIERA processes in a 'hard' way (command KILL) (opt.)
  4 - Disable scanning of data from SLCU to WS

<camera>    camera name, also root point in DB (default env. var. CCDNAME)
<WS env.>   name of workstation environment (default env. var. RTAPENV).
            If value is 0, no action on WS part of FIERA sw is taken
<SLCU env.> name of SLCU environment (default env. var. CCDLENV)
            If value is 0, no action on SLCU part of FIERA sw is taken
kill        kill all processes
```

ENVIRONMENT

```
CCDNAME  default for camera name (e.g. myccd)
RTAPENV  default for WS local environment (e.g. myws)
CCDLENV  default for SLCU environment (e.g myfiera)
```

RETURN VALUES

```
0 if SUCCESS
1 if FAILURE
```

CAUTIONS

```
The "kill" options should be used with care. By killing processes
'blindly', the system could remain in a dangerous state. To be used
only to recover when the system gets stuck.
```

EXAMPLES

```
> fcdDcsStop.sh myccd myws myfiera
      Terminate in a 'soft' way the FIERA sw both at WS and SLCU level
      for camera "myccd", WS environment "myws", SLCU environment "myfiera"

> fcdDcsStop.sh myccd myws myfiera kill
      Terminate in a 'hard' way the FIERA sw both at WS and SLCU level
      for camera "myccd", WS environment "myws", SLCU environment "myfiera"

> fcdDcsStop.sh myccd myws 0 kill
      Terminate in a 'hard' way the FIERA sw at WS level only
      for camera "myccd", WS environment "myws"

> fcdDcsStop.sh myccd 0 myfiera
      Terminate in a 'soft' way the FIERA sw at SLCU level only
      for camera "myccd", SLCU environment "myfiera"
```

SEE ALSO

```
      fcdDcsStart.sh, fcdDcsKill.sh



- - - - - -
Last change:  01/04/04-08:11
```

4.3.5 fcdDcsDbSave.sh

NAME

fcdDcsDbSave.sh - Save FIERA database values into backup file

SYNOPSIS

```
fcdDcsDbSave.sh [<camera>] [<env>] [<file>]
```

DESCRIPTION

```
This shell script reads some values from the FIERA WS database and
stores them into <file> using the utility dbBackup

<camera> camera name, also root point in DB (default env. var. CCDNAME)
<env>    environment where to read from (default env. var. RTAPENV)
<file>   name of backup file
         (default $INS_ROOT/SYSTEM/COMMON/CONFIGFILES/<camera>.dbcfg)
```

FILES

```
fcdConfig.inp     input file for utility dbBackup
```

ENVIRONMENT

```
CCDNAME   default for camera name (e.g. myccd)
RTAPENV   default environment (e.g. myws)
INS_ROOT  default for instrument root directory
INS_USER  default to SYSTEM if not defined
```

RETURN VALUES

```
0 if SUCCESS
1 if FAILURE
```

EXAMPLES

```
fcdDcsDbSave.sh myccd myws $INS_ROOT/$INS_USER/COMMON/CONFIGFILES/myccd.dbcfg
```

SEE ALSO

dbBackup

```
- - - - - -
Last change:  01/04/04-08:11
```

4.3.6 fcdDcsScan.sh

NAME

fcdDcsScan.sh - configure the DB values to be scanned from SLCU for

SYNOPSIS

```
fcdDcsScan.sh [<camera>] [<WS env.>] [<SLCU env.>] [<option>]
```

DESCRIPTION

```
This shell script configures the Scan System to retrieve DB values
for the FIERA SLCU to WS.

<camera>   camera name, also root point in DB (default env. var. CCDNAME)
<WS env.>  name of workstation environment (default env. var. RTAPENV)
<SLCU env.> name of SLCU environment (default env. var. CCDLENV)
<option>   a (default) - add entries to scan list
           c           - clean entries from scan list
           d           - disable scanning from the SLCU
           e           - enable scanning from the SLCU
```

ENVIRONMENT

```
CCDNAME  default for camera name (e.g. myccd)
RTAPENV  default for WS local environment (e.g. myws)
CCDLENV  default for SLCU environment (e.g. myfiera)
```

RETURN VALUES

```
0 if SUCCESS
1 if FAILURE
```

CAUTIONS

```
1 - It is assumed here that FIERA branch in the SLCU database is attached
    directly to the root point (not a sub-point of something else).
2 - If this script is run twice or more, the system may not behave properly
    any more. In this case better to shutdown the whole system and restart
    it again.
```

EXAMPLES

```
fcdDcsScan.sh myccd myws myfiera
     Add entries for camera "myccd", SLCU environment "myfiera",
     WS environment "myws"
fcdDcsScan.sh myccd myws myfiera c
     Clean entries for camera "myccd", SLCU environment "myfiera",
     WS environment "myws"
fcdDcsScan.sh 0 myws myfiera d
     Disable scanning from SLCU environment "myfiera" to
     WS environment "myws"
fcdDcsScan.sh 0 myws myfiera e
     Enable scanning from SLCU environment "myfiera" to
     WS environment "myws"
```

```
- - - - - -
Last change:  01/04/04-08:11
```

4.3.7 fcdosScan.sh

NAME

fcdosScan.sh - configure the DB values to be scanned from SLCU for CCD OS

SYNOPSIS

fcdosScan.sh [<camera>] [<WS env.>] [<SLCU env.>] [<option>]

DESCRIPTION

```
This shell script configures the Scan System to retrieve DB values
from the FIERA SLCU to WS (stand-alone part only).

<camera>    camera name, also root point in DB (default env. var. CCDNAME)
<WS env.>   name of workstation environment (default env. var. RTAPENV)
<SLCU env.> name of SLCU environment (default env. var. CCDLENV)
<option>    a (default) - add entries to scan list
            c           - clean entries from scan list
            d           - disable scanning from the SLCU
            e           - enable scanning from the SLCU
```

ENVIRONMENT

```
CCDNAME  default for camera name (e.g. myccd)
RTAPENV  default for WS local environment (e.g. myws)
CCDLENV  default for SLCU environment (e.g myfiera)
```

RETURN VALUES

```
0 if SUCCESS
1 if FAILURE
```

CAUTIONS

```
1 - It is assumed here that CCD branch in the SLCU database is attached
    directly to the root point (not a sub-point of something else).
2 - If this script is run twice or more, the system may not behave properly
    any more. In this case better to shutdown the whole system and restart
    it again.
```

EXAMPLES

```
fcdosScan.sh myccd myws myfiera
     Add entries for camera "myccd", SLCU environment "myfiera",
     WS environment "myws"
fcdosScan.sh myccd myws myfiera c
     Clean entries for camera "myccd", SLCU environment "myfiera",
     WS environment "myws"
fcdosScan.sh myccd myws myfiera e
     Enable entries for camera "myccd", SLCU environment "myfiera",
     WS environment "myws"
```

SEE ALSO

fcdDcsScan.sh

```
- - - - - -
Last change:  01/04/04-08:11
```

4.3.8 fcdDcsWcs.sh

NAME

```
fcdDcsWcs.sh - Set World Coordinate values in OLDB for CCD camera
```

SYNOPSIS

```
fcdDcsWcs.sh <camera> <WS env.> <xrefpix> <yrefpix>
             <secpix> <rotate> <equinox> <epoch> <proj>
```

DESCRIPTION

```
This shell script writes in the WS FIERA OLDB the values of attributes
related to the display of World Coordinates with rtd.
It is meant as a utility to applications having to configure the
WCS parameters for the camera used.

<camera>     camera name, also root point in DB
<WS env.>    name of workstation environment
<xrefpix>    X Coordinate of Reference Pixel
<yrefpix>    Y Coordinate of Reference Pixel
<secpix>     Number of arcseconds per pixel
<rotate>     Rotation angle (clockwise positive) in degrees
<equinox>    Equinox of coordinates, 1950 and 2000 supported
<epoch>      Epoch of coordinates, used for FK4/FK5 conversion
<proj>       Projection
```

RETURN VALUES

```
0 if SUCCESS
1 if FAILURE
```

EXAMPLES

```
> fcdDcsWcs.sh myccd myws 200.0 145.0 0.3 0.0 1950 0.0 PIXEL
      Reference pixel [200.0;145.0]
      Each pixel covers 0.3 arcseconds
      Rotation angle is 0.0 degrees
      Equinox of ccordinates set to 1950
      Epoch of ccordinates set to 0.0
      Projection set to PIXEL
```

SEE ALSO

```
rtdImageEvent.h for more information about parameters meaning and
                possible values
```

```
- - - - - -
Last change:  01/04/04-08:11
```

4.3.9 fcdDcsSetLogMask.sh

NAME

fcdDcsSetLogMask.sh - Set FITS logging mask in OLDB for CCD camera

SYNOPSIS

fcdDcsSetLogMask.sh <camera> <WS env.> <logMask>

DESCRIPTION

This shell script writes in the WS CCD OLDB the value of the attribute related to the FITS logging of main info.
An update of the $INS_ROOT/SYSTEM/COMMON/CONFIGFILES/<camera>.dbcfg file is performed. As well a snap of the WS CCD OLDB is made.
It is meant as a utility to applications having to configure the logMask parameter for the camera used.

<camera> camera name, also root point in DB
<WS env.> name of workstation environment
<logMask> Mask for FITS logging

RETURN VALUES

0 if SUCCESS
1 if FAILURE

EXAMPLES

```
> fcdDcsSetLogMask.sh myccd myws UT1AGA
      FITS Logs [UT1AGA]
```

- - - - - -
Last change: 01/04/04-08:11

4.3.10 fcdDcsSlcuHalt.sh

NAME

fcdDcsSlcuHalt.sh - halt FIERA SLCU

SYNOPSIS

fcdDcsSlcuHalt.sh [<SLCU env.>]

DESCRIPTION

This shell script performs a halting of FIERA SLCU.

<SLCU env.> name of SLCU environment (default env. var. CCDLENV)
If value is 0, no action on FIERA SLCU is taken

ENVIRONMENT

CCDLENV default for SLCU environment (e.g myfiera)

RETURN VALUES

0 if SUCCESS
1 if FAILURE

EXAMPLES

> fcdDcsSlcuHalt.sh myfiera
Halt the FIERA SLCU defined by the "myfiera" environment

> fcdDcsSlcuHalt.sh
Halt the FIERA SLCU defined by the CCDLENV environment

- - - - - -
Last change: 01/04/04-08:11

4.3.11 fcdDcsSlcuReboot.sh

NAME

fcdDcsSlcuReboot.sh - Reboots FIERA DCS SLCU

SYNOPSIS

fcdDcsSlcuReboot.sh [<camera>] [<WS env.>] [<SLCU env.>] [<option>]

DESCRIPTION

This shell script reboots the FIERA SLCU, independently from what is going on there.

<camera> camera name, also root point in DB (default env. var. CCDNAME)
<WS env.> name of workstation environment (default env. var. RTAPENV).
If value is 0, no action on WS part of FIERA sw is taken
<SLCU env.> name of SLCU environment (default env. var. CCDLENV)
If value is 0, no action on SLCU part of FIERA sw is taken
<option> s - restart also the scan system

CAUTIONS

This script should be used with care. If anything is going on in the SLCU it will get lost !!!!

EXAMPLES

> fcdDcsSlcuReboot.sh myccd 0 myfiera
Reboot SLCU defined by the "myfiera" environment, used for camera "myccd"

> fcdDcsSlcuReboot.sh myccd myws myfiera s
Reboot SLCU defined by the "myfiera" environment, used for camera "myccd" and restart the scan system

SEE ALSO

fcdDcsScan.sh

- - - - - -
Last change: 01/04/04-08:11

4.4 GUI classes

The following GUI classes, built with the VLT panel editor (see [12]) for being used within bigger application panels, are included in the library libfcdGuiPublic.tcl:

1. **fcdExpStatus_uifClass**. It displays the status of an exposure.
2. **fcdIpStatus_uifClass**. It displays the status of the real-time image processing on SLCU.
3. **fcdExpSetup_uifClass**. It displays the main setup definitions of the running exposure.
4. **fcdReadoutSetup_uifClass**. It displays the readout setup definitions of the running exposure.

4.4.1 fcdExpStatus_uifClass

NAME

```
fcdExpStatus_uifClass - GUI class for CCD exposure status display

Application panels who intend to use this class, must register to the
library libfcdGuiPublic.tcl.
All database attributes defined within this class are relative to the
root point of the camera used. Therefore, when importing an instance of
this class in a panel, the Database Current Working Point must be set
accordingly. For example, if the camera used is called "myccd", then
the CWP has to be <alias>myccd.

Note 1: The class shows the contents of the Workstation on-line database.
        The CCS Scan System must be enabled and working in order to
        retrieve the correct values from the FIERA SLCU.

Note 2: The class works only with exposures under the point
        exposures:exposure_1
```

APPLICATION AREA

```
This class displays only output fields.
In the order from top to button, left to right:

"Exposure status"
    String showing the current exposure status (updated on change).

"Remaining time (sec)"
    Remaining integration time (sec) (updated on polling).

"Loop count"
    Current repetition of the defined exposure (updated on polling).
```

BUGS

```
The attributes updated with a polling mechanism have a fixed polling rate.
The change of values, although faster in the database, might be shown on
the panel with some delay (up to 2 seconds).



- - - - - -
Last change:  01/04/04-08:11
```

4.4.2 fcdIpStatus_uifClass

NAME

fcdIpStatus_uifClass - GUI class for CCD image processing status display

Application panels who intend to use this class, must register to the library libfcdGuiPublic.tcl.
All database attributes defined within this class are relative to the point images:process relative to the root point of the camera used.
Therefore, when importing an instance of this class in a panel, the Database Current Working Point must be set accordingly.
For example, if the camera used is called "myccd", then the CWP has to be <alias>myccd:images:process.

Note: The class shows the contents of the Workstation on-line database. The CCS Scan System must be enabled and working in order to retrieve the correct values from the CCD SLCU.

APPLICATION AREA

This class displays only output fields.
In the order from top to button, left to right:

"Min"
Minimum pixel value over the image (updated on polling).

"Min"
Maximum pixel value over the image (updated on polling).

"RMS"
RMS value over the image (updated on polling).

"Intensity"
Intensity of the pixel closest to the centroid position computed on SLCU. If set to 0, it indicates that no object has been detected. (updated on polling).

BUGS

The attributes updated with a polling mechanism have a fixed polling rate.
The change of values, although faster in the database, might be shown on the panel with some delay (up to 2 seconds).

- - - - - -
Last change: 01/04/04-08:11

4.4.3 fcdExpSetup_uifClass

NAME

```
fcdExpSetup_uifClass - GUI class for CCD exposure setup display

Application panels who intend to use this class, must register to the
library libfcdGuiPublic.tcl.
All database attributes defined within this class are relative to the
root point of the camera used.
Therefore, when importing an instance of this class in a panel, the
Database Current Working Point must be set accordingly.
For example, if the camera used is called "myccd", then
the CWP has to be <alias>myccd.

Note: The class shows the contents of the Workstation on-line database.
      The CCS Scan System must be enabled and working in order to
      retrieve the correct values from the FIERA SLCU.
```

APPLICATION AREA

```
This class displays only output fields.
In the order from top to button, left to right:

"Time(sec)"
    Exposure time in seconds as defined by the user

"Repetitions"
    Number of times the same exposure has to be repeated. A value of 0
    means that the exposure has to be repeated forever until a STOP
    command is issued

"File"
    File where the last image has been saved.



- - - - - -
Last change:  01/04/04-08:11
```

4.4.4 fcdReadoutSetup_uifClass

NAME

```
fcdReadoutSetup_uifClass - GUI class for CCD readout setup display

Application panels who intend to use this class, must register to the
library libfcdGuiPublic.tcl.
All database attributes defined within this class are relative to the
root point of the camera used.
Therefore, when importing an instance of this class in a panel, the
Database Current Working Point must be set accordingly.
For example, if the camera used is called "myccd", then
the CWP has to be <alias>myccd.

Note: The class shows the contents of the Workstation on-line database.
      The CCS Scan System must be enabled and working in order to
      retrieve the correct values from the FIERA SLCU.
```

APPLICATION AREA

```
This class displays only output fields.
In the order from top to button, left to right:

"Speed"
    Readout speed as defined by the user

"Binning"
    "X"
    Horizontal binning factor as defined by the user
    "Y"
    Vertical binning factor as defined by the user
"Window"
    "#"
    Window index followed by field indicating if the window is enabled
    (YES) or not (NO)
    "DimX"
    Horizontal window size
    "DimY"
    Vertical window size
    "FirstX"
    Horizontal coordinate of window lower left corner
    "FirstY"
    Vertical coordinate of window lower left corner
```

```
- - - - - -
Last change:  01/04/04-08:11
```

4.5 Include files

All definitions needed by applications using the CCD software are contained in the file ***fcd.h***.
It includes the files ***fcdDbPublic.h***, which contains definitions of public database points/attributes, and ***fcdErrors.h***, which contains the CCD error codes.
These files only are considered public and **must** be used by external applications.

Other include files are installed as well, since they are needed by more than one CCD software module; nevertheless they are considered as private files to the CCD software: external applications are **not** allowed to use information contained in them, and therefore they are not documented in this section.

The public include files ***fcd.h***, ***fcdDbPublic.h*** and ***fcdErrors.h*** can be found in `$VLTROOT/include/` (or `$INTROOT/include`, if `INTROOT` is used).

4.6 Database

This section contains examples (to be used as templates) of:

1. DATABASE.db file
2. USER.db file

4.6.1 Example for DATABASE.db file

```
//*************************************************************************
//* E.S.O. - VLT project
//*
//* "@(#) $Id: DATABASE.db,v 1.12 2004/01/27 14:22:14 vltsccm Exp $"
//*
//* who       when      what
//* --------  --------  ----------------------------------------------
//* ccumani   11/11/95  Created
//*
//************************************************************************
//*
//*-----------------------------------------------------------------------
//

//************************************************************************
//*
//*  Template of DATABASE.db file for the FIERA database on the SPARC SLCU
//*  and the INS Workstation
//*
//*  This file contains CCS points that need to be customized to match
//*  the actual configuration.
//*
//*  The comments are guiding you to what shall/can be changed.
//*-----------------------------------------------------------------------


//*************************************************************
//    CCS DEFINITIONS
//*************************************************************

// Loads classes definition from standard file
#include "CCS.db"

//
// Loads points for CCS,
//
```

```
//     REMEMBER: to edit the provided template to match the
//               actual configuration
//
#include "USER.db"


//*************************************************************
//    APPLICATION DEFINITIONS
//*************************************************************

// Replace "<CCDNAME>" with the camera name
#define CCDNAME <CCDNAME>
#define CCDROOT :Appl_data:<CCDNAME>
#include "fcd.db"

// If you want a second CCD:
// 1) Replace "<CCDNAME2>" with the camera name
// 2) Uncomment next lines
//#undef CCDNAME
//#undef CCDROOT
//#define CCDNAME <CCDNAME2>
//#define CCDROOT :Appl_data:<CCDNAME2>
//#include "fcd.db"


//
// ___oOo___
```

4.6.2 Example for USER.db file

```
//************************************************************************
//* E.S.O. - VLT project
//*
//* "@(#) $Id: USER.db,v 1.12 2004/01/27 14:22:14 vltsccm Exp $"
//*
//* who       when      what
//* --------  --------  ----------------------------------------------
//* ccumani   11/11/95  Created
//* ccumani   20/03/00  "wmyccd" replaced by <WSENV> in SCAN SYSTEM points
//*
//*
//************************************************************************
//*
//*  Template of USER.db file for the FIERA database on the SPARC SLCU
//*  and the INS Workstation
//*
//*  This file contains CCS points that need to be customized to match
//*  the actual configuration.
//*
```

```
//*  The comments are guiding you to what shall/can be changed.
//*----------------------------------------------------------------------

// Loads classes definition from standard file
#include "CCS.db"

//
//************************************************************************
//   SCAN SYSTEM points
//************************************************************************
//

//
// - On the Workstation replace hereafter "<WSENV>" by the name of
//   of the SLCU environment the WS shall scan from
// - On the SPARC SCLU replace hereafter "<WSENV>" by the name of
//   the WS environment the SLCU shall report to
//
POINT "<VLT scan dev>" "ccs_config:scan config:LAN:<WSENV>"
BEGIN
        ALIAS   "<WSENV>"
END

//
//************************************************************************
//   SAMPLING PLOTTING  points
//************************************************************************
//
// If you need more plots, just duplicate this point, giving:
// - new point name
// - new alias
//

POINT PLOT  "ccs_config:plot config:samplePlot"
BEGIN
        ALIAS "samplePlot"
END

//
// ___oOo___
```

4.7 Setup files

This section contains an example of a complete FIERA CCD DCS file.

4.7.1 Example for complete FIERA CCD DCS setup file

```
#*********************************************************************
# E.S.O. - VLT project
#
# "@(#) $Id: fcdSetupComplete.det,v 2.84 2004/03/25 08:37:05 vltsccm Exp $"
#
# who       when      what
```

```
# --------  --------  ------------------------------------------
# abalestr  09/07/97  created (ported from CCD software)
# rdonalds  16/06/98  Added new keywords for centroiding
# rdonalds  19/08/98  Added IPLLX, LLY ..
# ccumani   07/09/98  description for DET.WINi.RMS corrected
# ccumani   08/09/98  SUP keywords substituted with PAF
# ccumani   09/09/98  - DET.WIN1.NDIT, DET.WIN1.ASUIT1,
#                       DET.WIN<i>.BACKGND, DET.WIN<i>.THRMIN,
#                       DET.READ.SHIFT1, DET.READ.SHIFTYP added
#                     - keyword order reorganised, comments added
#

#
# PAF Header
# ----------
PAF.HDR.START;                  # Start of PAF Header
PAF.TYPE     "Detector Setup"; # Type of PAF
PAF.ID              "        "; # ID for PAF
PAF.NAME            "        "; # Name of PAF
PAF.DESC            "        "; # Short description of PAF
PAF.CRTE.NAME       "        "; # Name of creator
PAF.CRTE.DAYTIM     "        "; # Civil Time for creation
PAF.LCHG.NAME       "        "; # Name of person/appl. changing
PAF.LCHG.DAYTIM     "        "; # Timestamp of last change
PAF.CHCK.NAME       "        "; # Name of appl. checking
PAF.HDR.END;                    # End of PAF Header


#
# Exposure definitions
# --------------------
DET.EXP.TYPE       "Normal  "; # Type of exposure as known to the CCD sw
DET.EXP.NREP                1; # number of repeated exposures
DET.EXP.TIMEREP          0.00; # Interval between two repeated exposures
DET.FRAM.TYPE      "Normal  "; # Type of frame
DET.WIN1.UIT1            1.000; # user defined subintegration time

# if "Multiple" exposure :
DET.WIN1.NDIT               3; # # of subintegrations
DET.WIN1.UIT2           2.000; # user defined subintegration time
DET.WIN1.UIT3           1.000; # user defined subintegration time
DET.WIN1.ASUIT1           "F"; # All UITi as UIT1 else as defined in UITi
DET.READ.SHIFT1            10; # Lines shifted between integration
DET.READ.SHIFTYP       "idem"; # Line shift type

# readout mode and binning :
DET.READ.CLKIND             1; # Index of clock pattern used
DET.WIN1.BINX               1; # Binning factor along X
DET.WIN1.BINY               1; # Binning factor along Y

# image transmission and storage :
DET.FRAM.FITSMTD            2; # How image data have to be saved on disk
DET.FRAM.FITSUNC "myImage.fits"; # Name of disk file for uncompressed data
DET.DISPLAY                 0; # real-time image display frame ID
DET.FRAM.SAMPLE             1; # Image sampling on workstation


#
```

```
# Window 1
# --------

# definition :
DET.WIN1.ST               "F"; # If T, window enabled
DET.WIN1.STRX             200; # Lower left pixel in X
DET.WIN1.NX                20; # # of pixels along X
DET.WIN1.STRY             300; # Lower left pixel in Y
DET.WIN1.NY                20; # # of pixels along Y

# image analysis :
DET.WIN1.MINMAX           "F"; # image extrema search performed
DET.WIN1.RMS                0; # image RMS calculation performed
DET.WIN1.CENTROID      "none"; # type of centroiding calculation
DET.WIN1.BACKGND            0; # sky background
DET.WIN1.THRMIN             0; # lower threshold
DET.WIN1.IPFUNC        "none"; # function for image analysis
DET.WIN1.IPLLX              0; # X offset from LL for image processing
DET.WIN1.IPLLY              0; # Y offset from LL for image processing
DET.WIN1.IPURX              0; # X offset from UR for image processing
DET.WIN1.IPURY              0; # Y offset from UR for image processing
DET.WIN1.REFX          210.00; # x position of reference star for centroiding
DET.WIN1.REFY          310.00; # y position of reference star for centroiding

#
# Window 2
# --------

# definition :
DET.WIN2.ST               "F"; # If T, window enabled
DET.WIN2.STRX               1; # Lower left pixel in X
DET.WIN2.NX                20; # # of pixels along X
DET.WIN2.STRY               1; # Lower left pixel in Y
DET.WIN2.NY                20; # # of pixels along Y

# image analysis :
DET.WIN2.MINMAX           "F"; # image extrema search performed
DET.WIN2.RMS                0; # image RMS calculation performed
DET.WIN2.CENTROID      "none"; # type of centroiding calculation
DET.WIN2.BACKGND            0; # sky background
DET.WIN2.THRMIN             0; # lower threshold
DET.WIN2.IPFUNC        "none"; # function for image analysis
DET.WIN2.IPLLX              0; # X offset from LL for image processing
DET.WIN2.IPLLY              0; # Y offset from LL for image processing
DET.WIN2.IPURX              0; # X offset from UR for image processing
DET.WIN2.IPURY              0; # Y offset from UR for image processing
DET.WIN2.REFX           10.00; # x position of reference star for centroiding
DET.WIN2.REFY           10.00; # y position of reference star for centroiding

# --- oOo ---
```

4.8 Image files

This section contains examples of:

1. FITS standard keywords written by the FIERA CCD DCS in the FITS header,
2. FITS hierarchical keywords written by the FIERA CCD DCS in a separate file,
3. FITS standard and hierarchical keywords written by the FIERA CCD DCS in the FITS header, when the " image extension per port"format (see 2.14.3.1) is chosen,
4. FITS hierarchical keywords written by the FIERA CCD DCS in a separate file, when the " image extension per port"(see 2.14.3.1) format is chosen.

4.8.1 Example of FITS standard keywords

In the following example, a mosaic of 2 chips is read via 2 outputs (one for each chip):

```
SIMPLE  =                    T / Standard FITS format (NOST-100.0)
BITPIX  =                   16 / # of bits storing pix values
NAXIS   =                    2 / # of axes in frame
NAXIS1  =                 1428 / # pixels/axis
NAXIS2  =                 1365 / # pixels/axis
ORIGIN  = 'ESO    '            / European Southern Observatory
DATE    = '1998-03-25T12:48:06.935'  / UT date when this file was written
CRVAL1  =                  1.0 / value of X ref pixel
CRPIX1  =                  1.0 / X Ref. pixel of center of rotation
CDELT1  =                  3.0 / Binning factor
CTYPE1  = 'PIXEL  '            / Pixel coordinate system
CRVAL2  =                  1.0 / value of X ref pixel
CRPIX2  =                  1.0 / X Ref. pixel of center of rotation
CDELT2  =                  3.0 / Binning factor
CTYPE2  = 'PIXEL  '            / Pixel coordinate system
BSCALE  =                  1.0 / pixel=FITS*BSCALE+BZERO
BZERO   =              32768.0 / pixel=FITS*BSCALE+BZERO
MJD-OBS =      50897.53325434  / MJD start (1998-03-25T12:47:53.175)
DATE-OBS= '1998-03-25T13:47:53.175'  / Date of observation
EXPTIME =              21.8394 / Total integration time
EXTEND  =                    F / Extension may be present
```

4.8.2 Example of FITS hierarchical keywords

```
HIERARCH ESO DET ID         = 'CCD FIERA - Rev 2.10' / Detector system Id
HIERARCH ESO DET NAME       = 'ccdsusi - ccdsusi' / Name of detector system
HIERARCH ESO DET DATE       = '1998-01-28' / Installation date
HIERARCH ESO DET DID        = 'ESO-VLT-DIC.CCDDCS,ESO-VLT-DIC.FCDDCS' / Ddictio
HIERARCH ESO DET BITS       =          16 / Bits per pixel readout
HIERARCH ESO DET SOFW MODE  = 'Normal  ' / CCD sw operational mode
HIERARCH ESO DET CHIPS      =           2 / # of chips in detector array
HIERARCH ESO DET CHIP1 ID   = 'ccd45   ' / Detector chip identification
HIERARCH ESO DET CHIP1 NAME = 'EEV44-80' / Detector chip name
HIERARCH ESO DET CHIP1 DATE = '1998-01-28' / Date of installation [YYYY-MM-DD]
HIERARCH ESO DET CHIP1 X    =           1 / X location in array
HIERARCH ESO DET CHIP1 Y    =           1 / Y location in array
HIERARCH ESO DET CHIP1 NX   =        2048 / # of pixels along X
```

```
HIERARCH ESO DET CHIP1 NY    =        4096 / # of pixels along Y
HIERARCH ESO DET CHIP1 PSZX  =        15.0 / Size of pixel in X
HIERARCH ESO DET CHIP1 PSZY  =        15.0 / Size of pixel in Y
HIERARCH ESO DET CHIP2 ID    = 'ccd46  '   / Detector chip identification
HIERARCH ESO DET CHIP2 NAME  = 'EEV44-80'  / Detector chip name
HIERARCH ESO DET CHIP2 DATE  = '1998-01-28' / Date of installation [YYYY-MM-DD]
HIERARCH ESO DET CHIP2 X     =           2 / X location in array
HIERARCH ESO DET CHIP2 Y     =           1 / Y location in array
HIERARCH ESO DET CHIP2 NX    =        2048 / # of pixels along X
HIERARCH ESO DET CHIP2 NY    =        4096 / # of pixels along Y
HIERARCH ESO DET CHIP2 PSZX  =        15.0 / Size of pixel in X
HIERARCH ESO DET CHIP2 PSZY  =        15.0 / Size of pixel in Y
HIERARCH ESO DET EXP ID      =          21 / Unique exposure ID number
HIERARCH ESO DET EXP TYPE    = 'Normal '   / Exposure type
HIERARCH ESO DET EXP DUMDIT  =           0 / # of dummy readouts
HIERARCH ESO DET EXP RDTTIME =       6.815 / image readout time
HIERARCH ESO DET EXP XFERTIM =      29.675 / image transfer time
HIERARCH ESO DET READ MODE   = 'normal '   / Readout method
HIERARCH ESO DET READ SPEED  = 'normal '   / Readout speed
HIERARCH ESO DET READ CLOCK  = 'readout L' / Readout clock pattern used
HIERARCH ESO DET OUTPUTS     =           2 / # of outputs
HIERARCH ESO DET OUTREF      =           0 / reference output
HIERARCH ESO DET OUT1 ID     = 'L      '   / Output ID as from manufacturer
HIERARCH ESO DET OUT1 NAME   = 'L      '   / Description of output
HIERARCH ESO DET OUT1 CHIP   =           1 / index of chip it belongs to
HIERARCH ESO DET OUT1 X      =           1 / X location of output
HIERARCH ESO DET OUT1 Y      =           1 / Y location of output
HIERARCH ESO DET OUT1 NX     =        2048 / valid pixels along X
HIERARCH ESO DET OUT1 NY     =        4096 / valid pixels along Y
HIERARCH ESO DET OUT1 PRSCX  =          50 / Prescan region in X
HIERARCH ESO DET OUT1 OVSCX  =          46 / Overscan region in X
HIERARCH ESO DET OUT1 CONAD  =        2.24 / Conversion from electrons to ADUs
HIERARCH ESO DET OUT1 GAIN   =        0.45 / Gain for output
HIERARCH ESO DET OUT3 ID     = 'L      '   / Output ID as from manufacturer
HIERARCH ESO DET OUT3 NAME   = 'L      '   / Description of output
HIERARCH ESO DET OUT3 CHIP   =           2 / index of chip it belongs to
HIERARCH ESO DET OUT3 X      =        2049 / X location of output
HIERARCH ESO DET OUT3 Y      =           1 / Y location of output
HIERARCH ESO DET OUT3 NX     =        2048 / valid pixels along X
HIERARCH ESO DET OUT3 NY     =        4096 / valid pixels along Y
HIERARCH ESO DET OUT3 PRSCX  =          50 / Prescan region in X
HIERARCH ESO DET OUT3 OVSCX  =          46 / Overscan region in X
HIERARCH ESO DET OUT3 CONAD  =        2.26 / Conversion from electrons to ADUs
HIERARCH ESO DET OUT3 GAIN   =        0.44 / Gain for output
HIERARCH ESO DET FRAM ID     =           1 / Image sequencial number
HIERARCH ESO DET FRAM TYPE   = 'Normal '   / Type of frame
HIERARCH ESO DET WINDOWS     =           1 / # of windows readout
HIERARCH ESO DET WIN1 STRX   =           1 / Lower left pixel in X
HIERARCH ESO DET WIN1 STRY   =           1 / Lower left pixel in Y
HIERARCH ESO DET WIN1 NX     =        1428 / # of pixels along X
HIERARCH ESO DET WIN1 NY     =        1365 / # of pixels along Y
HIERARCH ESO DET WIN1 BINX   =           3 / Binning factor along X
HIERARCH ESO DET WIN1 BINY   =           3 / Binning factor along Y
```

HIERARCH ESO DET WIN1 NDIT = 1 / # of subintegrations
HIERARCH ESO DET WIN1 UIT1 = 30.000 / user defined subintegration time
HIERARCH ESO DET WIN1 DIT1 = 21.8394 / actual subintegration time
HIERARCH ESO DET WIN1 DKTM = 36.3018 / Dark current time
HIERARCH ESO DET SHUT TYPE = 'Slit ' / type of shutter
HIERARCH ESO DET SHUT ID = 'ccdsusi shutter' / Shutter unique identifier
HIERARCH ESO DET SHUT TMOPEN = 0.000 / Time taken to open shutter
HIERARCH ESO DET SHUT TMCLOS = 0.000 / Time taken to close shutter
HIERARCH ESO DET TELE INT = 1.0 / Interval between two successive te
HIERARCH ESO DET TELE NO = 2 / # of sources active
HIERARCH ESO DET TLM0 NAME = 'CCD Temp' / Description of telemetry param.
HIERARCH ESO DET TLM0 ID = 'CCD Temp' / ID of telemetry sensor
HIERARCH ESO DET TLM0 START = 273.10 / Telemetry value at read start
HIERARCH ESO DET TLM0 END = 273.20 / Telemetry value at read completion
HIERARCH ESO DET TLM1 NAME = 'Elec Temp' / Description of telemetry param.
HIERARCH ESO DET TLM1 ID = 'Elec Temp' / ID of telemetry sensor HIERARCH ESO DET TLM1 START = 300.41 / Telemetry value at read start
HIERARCH ESO DET TLM1 END = 300.52 / Telemetry value at read completion

4.8.3 Example of FITS standard keywords (Image Extension per Port)

In the following example, a mosaic of 2 chips is read via 2 outputs (one for each chip):

```
SIMPLE  =                    T          / Standard FITS format (NOST-100.0)
BITPIX  =                   16          / # of bits storing pix values
NAXIS   =                    0          / # of axes in frame
ORIGIN  = 'ESO     '                    / European Southern Observatory
DATE    = '1998-06-24T17:01:20.042'     / UT date when this file was written
MJD-OBS =       50988.62516799          / MJD start (1998-06-24T15:00:14.515)
DATE-OBS= '1998-06-24T17:00:14.514'     / Date of observation
EXPTIME =               0.0443          / Total integration time
EXTEND  =                    T          / Extension may be present



END


XTENSION= 'IMAGE   '                    / Extension first keyword (see SIMPLE)
BITPIX  =                   16          / # of bits storing pix values
NAXIS   =                    2          / # of axes in frame
NAXIS1  =                 2144          / # pixels/axis
NAXIS2  =                 4096          / # pixels/axis
PCOUNT  =                    0          / Number of parameters per group
GCOUNT  =                    1          / Number of groups
EXTNAME = 'WIN1.CHIP1.OUT1'             / Extension name
ORIGIN  = 'ESO     '                    / European Southern Observatory
DATE    = '1998-06-24T17:01:20.042'     / UT date when this file was written
CRVAL1  =                  1.0          / value of ref pixel
CRPIX1  =                 51.0          / Ref. pixel of center of rotation
CDELT1  =                  1.0          / Binning factor
CTYPE1  = 'PIXEL   '                    / Pixel coordinate system
CRVAL2  =                  1.0          / value of ref pixel
CRPIX2  =                  1.0          / Ref. pixel of center of rotation
CDELT2  =                  1.0          / Binning factor
```

```
CTYPE2  = 'PIXEL   '                   / Pixel coordinate system
BSCALE  =                  1.0         / pixel=FITS*BSCALE+BZERO
BZERO   =              32768.0         / pixel=FITS*BSCALE+BZERO
MJD-OBS =       50988.62516799         / MJD start (1998-06-24T15:00:14.515)
DATE-OBS= '1998-06-24T17:00:14.514'    / Date of observation
EXPTIME =               0.0443         / Total integration time
HIERARCH ESO DET CHIP INDEX  =           1 / Chip index
HIERARCH ESO DET CHIP ID     = 'ccd45   '   / Detector chip identification
HIERARCH ESO DET CHIP NAME   = 'EEV44-80'   / Detector chip name
HIERARCH ESO DET CHIP DATE   = '1998-01-28' / Date of installation [YYYY-MM-DD]
HIERARCH ESO DET CHIP X      =           1 / X location in array
HIERARCH ESO DET CHIP Y      =           1 / Y location in array
HIERARCH ESO DET CHIP NX     =        2048 / # of pixels along X
HIERARCH ESO DET CHIP NY     =        4096 / # of pixels along Y
HIERARCH ESO DET CHIP PSZX   =        15.0 / Size of pixel in X
HIERARCH ESO DET CHIP PSZY   =        15.0 / Size of pixel in Y
HIERARCH ESO DET CHIP XGAP   =   96.000000 / Gap between chips along x
HIERARCH ESO DET CHIP YGAP   =    0.000000 / Gap between chips along y
HIERARCH ESO DET OUT INDEX   =           1 / Output index
HIERARCH ESO DET OUT ID      = 'L       '   / Output ID as from manufacturer
HIERARCH ESO DET OUT NAME    = 'L       '   / Description of output
HIERARCH ESO DET OUT CHIP    =           1 / Chip to which the output belongs
HIERARCH ESO DET OUT X       =           1 / X location of output
HIERARCH ESO DET OUT Y       =           1 / Y location of output
HIERARCH ESO DET OUT NX      =        2048 / valid pixels along X
HIERARCH ESO DET OUT NY      =        4096 / valid pixels along Y
HIERARCH ESO DET OUT PRSCX   =          50 / Prescan region in X
HIERARCH ESO DET OUT OVSCX   =          46 / Overscan region in X
HIERARCH ESO DET OUT PRSCY   =           0 / Prescan region in Y
HIERARCH ESO DET OUT OVSCY   =           0 / Overscan region in Y
HIERARCH ESO DET OUT CONAD   =        2.24 / Conversion from ADUs to electrons
HIERARCH ESO DET OUT GAIN    =        0.45 / Conversion from electrons to ADU

END


[image data]


XTENSION= 'IMAGE   '                   / Extension first keyword (see SIMPLE)
BITPIX  =                   16         / # of bits storing pix values
NAXIS   =                    2         / # of axes in frame
NAXIS1  =                 2144         / # pixels/axis
NAXIS2  =                 4096         / # pixels/axis
PCOUNT  =                    0         / Number of parameters per group
GCOUNT  =                    1         / Number of groups
EXTNAME = 'WIN1.CHIP2.OUT1'            / Extension name
ORIGIN  = 'ESO     '                   / European Southern Observatory
DATE    = '1998-06-24T17:01:20.042'    / UT date when this file was written
CRVAL1  =                  1.0         / value of ref pixel
CRPIX1  =              -2093.0         / Ref. pixel of center of rotation
CDELT1  =                  1.0         / Binning factor
CTYPE1  = 'PIXEL   '                   / Pixel coordinate system
CRVAL2  =                  1.0         / value of ref pixel
CRPIX2  =                  1.0         / Ref. pixel of center of rotation
CDELT2  =                  1.0         / Binning factor
CTYPE2  = 'PIXEL   '                   / Pixel coordinate system
BSCALE  =                  1.0         / pixel=FITS*BSCALE+BZERO
```

```
BZERO   =              32768.0          / pixel=FITS*BSCALE+BZERO
MJD-OBS =       50988.62516799          / MJD start (1998-06-24T15:00:14.515)
DATE-OBS= '1998-06-24T17:00:14.514'     / Date of observation
EXPTIME =               0.0443          / Total integration time
HIERARCH ESO DET CHIP INDEX  =            2 / Chip index
HIERARCH ESO DET CHIP ID     = 'ccd46   '   / Detector chip identification
HIERARCH ESO DET CHIP NAME   = 'EEV44-80'   / Detector chip name
HIERARCH ESO DET CHIP DATE   = '1998-01-28' / Date of installation [YYYY-MM-DD]
HIERARCH ESO DET CHIP X      =            2 / X location in array
HIERARCH ESO DET CHIP Y      =            1 / Y location in array
HIERARCH ESO DET CHIP NX     =         2048 / # of pixels along X
HIERARCH ESO DET CHIP NY     =         4096 / # of pixels along Y
HIERARCH ESO DET CHIP PSZX   =         15.0 / Size of pixel in X
HIERARCH ESO DET CHIP PSZY   =         15.0 / Size of pixel in Y
HIERARCH ESO DET CHIP XGAP   =     0.000000 / Gap between chips along x
HIERARCH ESO DET CHIP YGAP   =     0.000000 / Gap between chips along y
HIERARCH ESO DET OUT INDEX   =            1 / Output index
HIERARCH ESO DET OUT ID      = 'L       '   / Output ID as from manufacturer
HIERARCH ESO DET OUT NAME    = 'L       '   / Description of output
HIERARCH ESO DET OUT CHIP    =            2 / Chip to which the output belongs
HIERARCH ESO DET OUT X       =         2049 / X location of output
HIERARCH ESO DET OUT Y       =            1 / Y location of output
HIERARCH ESO DET OUT NX      =         2048 / valid pixels along X
HIERARCH ESO DET OUT NY      =         4096 / valid pixels along Y
HIERARCH ESO DET OUT PRSCX   =           50 / Prescan region in X
HIERARCH ESO DET OUT OVSCX   =           46 / Overscan region in X
HIERARCH ESO DET OUT PRSCY   =            0 / Prescan region in Y
HIERARCH ESO DET OUT OVSCY   =            0 / Overscan region in Y
HIERARCH ESO DET OUT CONAD   =         2.26 / Conversion from ADUs to electrons
HIERARCH ESO DET OUT GAIN    =         0.44 / Conversion from electrons to ADU

END


[image data]
```

4.8.4 Example of FITS hierarchical keywords (Image Extension per Port)

```
HIERARCH ESO DET ID          = 'CCD FIERA - Rev 2.22 ' / Detector system Id
HIERARCH ESO DET NAME        = 'ccdtst1 - ccdsusi' / Name of detector system
HIERARCH ESO DET DATE        = '1998-01-28' / Installation date
HIERARCH ESO DET DID         = 'ESO-VLT-DIC.CCDDCS,ESO-VLT-DIC.FCDDCS' / Ddictio
HIERARCH ESO DET BITS        =           16 / Bits per pixel readout
HIERARCH ESO DET CHIPS       =            2 / # of chips in detector array
HIERARCH ESO DET OUTPUTS     =            4 / # of outputs
HIERARCH ESO DET OUTREF      =            0 / reference output
HIERARCH ESO DET WINDOWS     =            1 / # of windows readout
HIERARCH ESO DET SOFW MODE   = 'Embedded sw simulated' / CCD sw operational mode
HIERARCH ESO DET EXP ID      =           31 / Unique exposure ID number
HIERARCH ESO DET EXP TYPE    = 'Normal  '   / Exposure type
HIERARCH ESO DET EXP RDTTIME =        2.235 / image readout time
HIERARCH ESO DET EXP XFERTIM =       44.040 / image transfer time
HIERARCH ESO DET WIN1 ST     =            T / If T, window enabled
```

```
HIERARCH ESO DET WIN1 STRX   =            1 / Lower left pixel in X
HIERARCH ESO DET WIN1 STRY   =            1 / Lower left pixel in Y
HIERARCH ESO DET WIN1 NX     =         4288 / # of pixels along X
HIERARCH ESO DET WIN1 NY     =         4096 / # of pixels along Y
HIERARCH ESO DET WIN1 BINX   =            1 / Binning factor along X
HIERARCH ESO DET WIN1 BINY   =            1 / Binning factor along Y
HIERARCH ESO DET WIN1 NDIT   =            1 / # of subintegrations
HIERARCH ESO DET WIN1 UIT1   =        0.000 / user defined subintegration time
HIERARCH ESO DET WIN1 DIT1   =       0.0443 / actual subintegration time
HIERARCH ESO DET WIN1 DKTM   =       0.0443 / Dark current time
HIERARCH ESO DET READ MODE   = 'normal  '   / Readout method
HIERARCH ESO DET READ SPEED  = 'normal  '   / Readout speed
HIERARCH ESO DET READ CLOCK  = 'readout L'  / Readout clock pattern used
HIERARCH ESO DET FRAM ID     =            1 / Image sequencial number
HIERARCH ESO DET FRAM TYPE   = 'Normal  '   / Type of frame
```

4.9 Log files

This section contains examples of Configuration and operational logs in FITS format from a scientific CCD camera.

4.9.1 Example of FITS log from a Scientific CCD

```
15:14:52>-START DET SOFW / CCD control sw started [MSKSU]
15:14:52> DET SOFW ID = CCD Rev 2.10 / CCD control program name-version [MSKSU]
15:14:52> DET SOFW MODE = Normal / CCD sw operational mode [MSKSU]
15:14:52> DET  STATE = LOADED / CCD sw state [MSKSU]
15:14:54>-CHANGE DET STATE / CCD sw state change [MSKSU]
15:15:30> DET SHUT TYPE = Slit / type of shutter [MSKSU]
15:15:30> DET SHUT ID = ccdsusi shutter / Shutter unique identifier [MSKSU]
15:15:12> DET  STATE = STANDBY / CCD sw state [MSKSU]
15:15:12>-CHANGE DET STATE / CCD sw state change [MSKSU]
15:15:15>/World Coordinates: Enabled [MSKSU]
15:15:15> DET  STATE = ONLINE / CCD sw state [MSKSU]
15:15:52>-START DET EXP / Start exposure [MSKSU]
15:15:52> DET EXP ID = 11 / Unique exposure ID number [MSKSU]
15:15:52>-START DET CHIP WIPE / CCD start wipe action [MSKSU]
15:15:55> DET CHIP WIPE STATUS = DONE / CCD wipe completed [MSKSU]
15:15:55>-OPEN DET SHUT / CCD shutter open [MSKSU]
15:16:03>-PAUSE DET EXP / Pause exposure [MSKSU]
15:16:03>-CLOSE DET SHUT / CCD shutter closed [MSKSU]
15:16:08>-RESUME DET EXP / Continue exposure [MSKSU]
15:16:08>-OPEN DET SHUT / CCD shutter open [MSKSU]
15:16:18>-STOP DET EXP / Stop integration / loop [MSKSU]
15:16:18>-CLOSE DET SHUT / CCD shutter closed [MSKSU]
15:16:18>-START DET CHIP READ / start readout of CCD chip [MSKSU]
15:15:52>-START DET TRANS / CCD data transfer started [MSKSU]
15:16:25> DET CHIP READ STATUS = DONE / CCD chip readout completed [MSKSU]
15:15:59> DET TRANS STATUS = DONE / CCD data transfer completed [MSKSU]
```

15:16:56>-START DET EXP / Start exposure [MSKSU]
15:16:56> DET EXP ID = 21 / Unique exposure ID number [MSKSU]
15:16:56>-START DET CHIP WIPE / CCD start wipe action [MSKSU]
15:16:59> DET CHIP WIPE STATUS = DONE / CCD wipe completed [MSKSU]
15:16:59>-OPEN DET SHUT / CCD shutter open [MSKSU]
15:17:09>-CLOSE DET SHUT / CCD shutter closed [MSKSU]
15:17:11>-START DET CHIP READ / start readout of CCD chip [MSKSU]
15:16:45>-START DET TRANS / CCD data transfer started [MSKSU]
15:17:17> DET CHIP READ STATUS = DONE / CCD chip readout completed [MSKSU]
15:16:51> DET TRANS STATUS = DONE / CCD data transfer completed [MSKSU]
15:30:49>-CHANGE DET STATE / CCD sw state change [MSKSU]
15:30:50> DET STATE = OFF / CCD sw state [MSKSU]

4.10 Error messages files

The error definition file ***fcd_ERRORS*** can be found in `$VLTROOT/ERRORS/` (or `$INTROOT/ERRORS`, if `INTROOT` is used).

5 INSTALLATION GUIDE

The FIERA CCD sw is delivered as part of the VLT sw distribution kit.

This section describes step by step all what is needed to make the FIERA CCD sw ready to run and the test procedures to verify the correctness of the installation.

5.1 General

The FIERA CCD Control Software currently includes:

- the FIERA CCD Software (WS, SLCU, DSP code).
- the PULPO Control Software

The distribution kit contains all the sources needed to regenerate the software and can be installed with or without the real camera hardware (“ SLCU Simulated”level, i.e., no FIERA hardware).

5.1.1 Copyright

See [7].

5.2 Supported configuration

5.2.1 Hardware

The following components are needed by FIERA CCD systems, in addition to the VLT standard ones, listed in [7]:

WS

- one or more high capacity hard-disk(s) for image storage (at least 4 GBytes are recommended).

FIERA SLCU (see [5])

- a SPARC CPU Board,
- a Detector Electronics Interface Board,
- an ATM Board (optional).

FIERA Detector Electronics (see [5])

- one Communication Board,
- 1 to 4 Video Boards,
- 1 to 4 Clock Driver Boards,
- 1 to 4 Analogue Bias Boards.

5.2.2 Software

UNIX Operating System (see [7] for the types and versions supported).

VLT Common Software, installed with CCSLite (according to [7])

(optional) VLT Real Time Display, installed according to [16].

5.3 Contents

The FIERA CCD Control Software is delivered as part of the VLT sw release. See [7] for more information about the contents.

5.4 Problem Reporting/Change Request

See [7] for how to report problems/errors or suggested changes in software or documentation.

5.5 FIERASW Installation

The FIERA CCD Control Software is delivered as part of the VLT sw release: its installation is part of the general VLT Software installation (see [7] and [19]).

NOTE: Remember to update the VLT Software on **BOTH** the Instrument Workstation and the FIERA SPARC SLCU.

In addition to the FIERA CCD Control Software, the appropriate Instrument Configuration File and CCD clock patterns have to be installed on the Instrument Workstation (from where the FIERA SLCU usually mounts the `INS_ROOT` via nfs, see also 5.6.6).

The Instrument Configuration File and the CCD clock patterns are delivered separately form the VLT Software. They are provided for each different instrument inside a module named `fcd<InstrumentName>`. The `fcd<InstrumentName>` module can be retrieved:

1. using the `cmmCopy` utility,
2. via anonymous ftp or through the web. In this case the module is available in a compressed file named `fcd<InstrumentName>.<version>.tar.gz`. and available in the "`code/FIERA`" area (follow the steps described in [7]).

To install the Instrument Configuration File and the CCD clock patterns:

3. login as `vltmgr` (see [7]) or `Instrument User` (see 5.6.2)
4. go in the directory where you intend to copy the **`fcd<InstrumentName>`** module. Here we call it `<FIERASW_ROOT>`.

   ```
   > cd <FIERASW_ROOT>
   ```

5. download the **`fcd<InstrumentName>`** module using cmmCopy or anonymous ftp or the web (follow the steps described in [7]). If you have downloaded the module via anonymous ftp or the web, gunzip and untar the Instrument Module file:

   > gunzip fcd<InstrumentName>.<version>.tar.gz

   > tar xvf fcd<InstrumentName>.<version>.tar

6. set the current directory to the Instrument Module source directory

   ```
   > cd fcd<InstrumentName>/src
   ```

7. install the Instrument Configuration File and CCD clock patterns

   ```
   > make clean all install
   ```

5.6 Configuration of the Instrument workstation

After successful installation of the software, the following steps will take you through the configuration of a complete CCD system. A CCD system requires a WS environment and a FIERA CCD Controller. Although existing environments can be used, we suggest performing the configuration using new environments. Once you are familiar with that, the integration of the FIERA CCD software to an existing project is a straightforward process.

This section assumes that you master the process of creating/configuring environments, including the directory structure and available tools. (If not, please have a look to the configuration and verification section of [7]).

5.6.1 Environment variables.

Some environment variables are used on Workstation by the FIERA CCD sw and therefore must be defined; some of them, **marked with (*), are optional**: if they are not defined, the specified default value is taken.

HOST
name of the local Workstation. It has to be set to the value returned by the command *hostname*:

```
> setenv HOST ‘hostname‘
```

RTAPENV
name of the workstation CCS environment where CCD sw runs. Example:

```
> setenv RTAPENV wmyccd
```

INS_ROOT
name of the root directory where data dictionaries, setup and configuration files are stored/retrieved (see [3] for the full directory structure). At least the root directory must exist, all the sub-directories needed by the FIERA CCD sw, if not existing, will be created and populated in the next step (5.6.6).

CCDNAME
name of the CCD camera (**max. 7 characters**). Examples:

```
> setenv CCDNAME myccd          right
> setenv CCDNAME myccdred       wrong! (8 characters)
```

The alias name of the root point for the database CCD branch (see also point 5.6.2) **must be the same as the value of the environment variable `CCDNAME`**.

CCDLENV
name of the SLCU environment where CCD sw runs. Example:

```
> setenv CCDLENV wmyfiera
```

CCDDXF (*)
name of the WS host receiving CCD image data (see [13]). If not defined, default is the contents of the environment variable `HOST` (same LAN used for messages and data). Example:

```
> setenv CCDDXF myws
```

INS_HOST (*)
name of the Workstation hosting the disk containing the `$INS_ROOT` directory. If not defined, default is the local Workstation (see `$HOST` above). Example:

```
> setenv INS_HOST $HOST
```

INS_USER (*)
INS_ROOT branch for setup files search (see [3] for more details). If not defined, default is SYSTEM. Example:

```
> setenv INS_USER SYSTEM
```

CCDDID (*)
Name of Common Data Interface Dictionary to be used (excluded prefix ESO-VLT-DIC). If not defined, default is CCDDCS. Example:

```
> setenv CCDDID MYCCD
```

The Data Interface Dictionary file must be stored in $INS_ROOT/SYSTEM/Dictionary.

The default Dictionary file (ESO-VLT-DIC.CCDDCS) is in $VLTROOT/config.

FCDDID (*)
Name of FIERA Data Interface Dictionary to be used (excluded prefix ESO-VLT-DIC). If not defined, default is FCDDCS. Example:

```
> setenv FCDDID MYFIERA
```

The Data Interface Dictionary file must be stored in $INS_ROOT/SYSTEM/Dictionary.

The default Dictionary file (ESO-VLT-DIC.FCDDCS) is in $VLTROOT/config.

FCD_IT_HOST (mandatory only if "image extensions per port" - see 2.14.3.1 - are used)
Name of the SLCU HOST to which connect in order to transfer images when using image extension format. It MUST be defined, no default is available. Example:

```
> setenv FCD_IT_HOST remote_host
```

In the following sections we assume as example that the following definitions are done (all optional variables are defaulted):

```
> setenv CCDNAME myccd
> setenv RTAPENV wmyccd
```

> setenv HOST 'hostname'

```
> setenv CCDLENV wmyfiera
```

> setenv INS_ROOT /diskc/myName/insroot

5.6.2 Users to be defined

Some users must be defined on the Instrument Workstation to run the FIERA Software:

1. fcdrun.

   This is the special user account used by the FIERA Software on the Instrument Workstation for executing the CCD S/W on the SPARC (see also 5.7.3).

2. Instrument User.

   This is the user which runs the Fiera Software. In order for the OLSB to function correctly, this user must be defined on both the Instrument Workstation and the FIERA SLCU, with the same user id must (see also 5.7.4).

5.6.3 WS Environment

1. Create a new environment as in [7]. Remember, the environment shall be named as in `$RTAPENV`.
   Do not forget to properly configure `/etc/services`, `/etc/$RTAPROOT/etc/RtapEnvList` (or `$RTAPROOT/etc/RtapEnvList` on older Rtap releases), etc. The environment to be used on the SPARC must also be added to these files.
   a. start *ccsei*
      ```
      > ccsei &
      ```
   b. select "*CCS Environment Setup..*". The utility *vccEnv* is started.
   c. enter in the *vccEnv* panel the name of the environment to create
   d. Press the button *Create*
   e. Keep the panel *vccEnv* open: you will need in the next steps as well.
2. Configure the database
   a. set to the environment dbl directory:
      ```
      > cd $VLTDATA/ENVIRONMENTS/wmyccd/dbl
      ```
   b. Add the CCD to the database:
      i. edit DATABASE.db file (see also example in section 4.6.1).
      ii. move to application definitions section.
      iii. Add the following lines:
      ```
      #define CCDNAME myccd                // camera name
      #define CCDROOT :Appl_data:myccd     // CCD branch root point
      #include "fcd.db"                    // build CCD branch
      ```
   c. Add a dedicated point to the scan system
      i. edit USER.db file (see also example in section 4.6.2).
      ii. move to the scan system section.
      iii. Add the following lines (instead of lmyccd use your CCDLENV!)
      ```
      POINT "<VLT scan dev>" "ccs_config:scan config:LAN:wmyfiera"
      BEGIN
          Alias wmyfiera
      END
      ```
   d. regenerate the online database files:
      ```
      > make clean db
      ```
   e. verify the correctness of the generated files:
      ```
      > cd ../DB/Appl_data
      > ls
      ```
      ...if the CCD branch has been successfully created, a subdirectory *myccd* exists
3. generate the environment and then shutdown it:
   a. Press the button *Init* in the *vccEnv* panel
4. Start the environment:
   a. Press the button *Start* in the *vccEnv* panel
   b. Close the panel *vccEnv*: not needed any more (*File->Quit*)

c. Close the panel *ccsei*: not needed any more (*File->Quit*)

5.6.3.1 WS Environment Verification

1. Verify that the environment has been generated:

```
> dbRead "<alias>myccd.date"
```

The following reply should appear:

```
BYTES value =
```

2. Verify that the needed processes are running in the CCS environment (use *ccsPerfMon* for verification):

```
Process Name          |PNUM |  PID  |  UID  |  GID  |  MSGID  | MONPID
^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
ccsScheduler          | 1   | 17141 | 3227  | 300   | -1      |    -1
ccsSHManager          | 2   | 17145 | 3227  | 300   | 53110   |    -1
qsemu                 | 3   | 17161 | 3227  | 300   | 8709    |    -1
evtEventConfg         | 4   | 17177 | 3227  | 300   | 3623    |    -1
timsTimeKeeper        | 5   | 17163 | 3227  | 300   | 47404   |    -1
scanMngr              | 6   | 17169 | 300   | 300   | 34511   |    -1
ccsScan               | 7   | 17192 | 300   | 300   | 60026   |    -1
ccsCmdServer          | 8   | 17175 | 3227  | 300   | 3922    |    -1
alrmServer            | 9   | 17171 | 3227  | 300   | 32112   |    -1
logManager            | 13  | 17173 | 3227  | 300   | 4521    |    -1
msgServer             | 14  | 17179 | 3227  | 300   | 39924   |    -1
cmdManager            | 15  | 17181 | 3227  | 300   | 40825   |    -1
alrmLogger            | 16  | 17250 | 3227  | 300   | 24008   |    -1
hisDHMngr             | 17  | 17252 | 300   | 300   | 63331   | 17254
dbMQDBM               | 58  | 17165 | 3227  | 300   | 488102  |    -1
```

5.6.4 Real Time Display

CCD Software can work with or without the Real Time Display. If you have RTD installed, check that the process *rtdServer* is already running:

```
> ps -aef | grep rtdServer | grep -v grep
```

If this is not the case, start it:

```
> rtdServer &
```

NOTE: It is possible to start the *rtdServer* at boot, modifying the `$VLTDATA/ENVIRONMENTS/$CCDNAME/RtapEnvTable.normal` file (see [10]), adding a line which should look like the following:

```
 26  2  N  N  N  A  N  100  128   rtdServer
```

5.6.5 Setting source mask for FITS logs

As already said in section 2.16, the source mask for FITS log has to be defined, using the script *fcdDcsSetLogMask.sh* (see also section 3.3). Example:

```
> fcdDcsSetLogMask.sh $CCDNAME $RTAPENV "  UT1AGA"
```

5.6.6 Files needed for CCD operations

The Instrument Configuration File and CCD clock patterns[1] must be installed in the proper subdirectory of `INS_ROOT`; to do that, the FIERA CCD sw provides a UNIX shell script (*fcdInstall.sh*).
In order to run the *fcdInstall.sh* script, the `$INS_ROOT` directory has to be created:

```
> mkdir $INS_ROOT
```

The script needs as parameter the name of the file containing information about the specific camera used; there is one file for each CCD system; to list them all (see [7] for `VLTROOT`):

```
> ls $VLTROOT/config/fcd*.dbcfg
```

The choice of the right file for the system to be used should be straightforward if the following criteria are followed

- For the usual Scientific CCDs, the files have prefix *fcdSci<instrument{Arm}>.dbcfg (e.g. fcdSciUvesBlue.dbcfg)*. Example (from UNIX shell):

  ```
  > fcdInstall.sh fcdSciUvesBlue.dbcfg
  ```

  The template file *fcdSciTemplate.dbcfg* is available if no appropriate file is found.
- When tracker chips are used, the files have prefix *fcdTrk<instrument{Arm}>.dbcfg (e.g. fcdTrkTestCamera.dbcfg)*

  No template file for tracker chip *is available yet*.

For verification, do the following:

1. change working directory to $INS_ROOT

   ```
   > cd $INS_ROOT
   ```
2. Show contents of sub-directories:

   ```
   > ls -R
   ```

   At least the following directories must exist (see [3] for the meaning):

   $INS_ROOT/SYSTEM/DETDATA

   $INS_ROOT/SYSTEM/Dictionary

   This directory should contain at least the following files:

   ESO-VLT-DIC.CCDDCS

   ESO-VLT-DIC.FCDDCS

   $INS_ROOT/SYSTEM/COMMON/CONFIGFILES

   This directory should contain at least the following files:

   fcdDcsFixed.dbcfg

   fcdSciTemplate.dbcfg (*)

   <CCDNAME>.dbcfg

   $INS_ROOT/SYSTEM/COMMON/SETUPFILES

   $INS_ROOT/SYSTEM/COMMON/SETUPFILES/REF

   $INS_ROOT/SYSTEM/COMMON/SETUPFILES/DET

   This directory should contain at least the following file:

   fcdSetupComplete.det (*)

---

1. files containing information needed to operate a CCD camera, see also the beginning of paragraph 5.5

All files and directories listed above must be present to be able to run the system. They must not be deleted nor modified. The only exceptions are template files, marked with (*), which might be useful to the user, but are not strictly needed, and the file `<CCDNAME>.dbcfg`, which might be modified (not deleted!) to accommodate modifications in the camera configuration (see 8.1).

Note: if the directory `$INS_ROOT/SYSTEM/COMMON/CONFIGFILES/$CCDNAME` and the file `$INS_ROOT/SYSTEM/COMMON/CONFIGFILES/$CCDNAME.dbcfg` already exist, they are first saved as backup and then overwritten. The old information, if changed by the user, must then be re-entered (see 8.1).

5.6.7 Scan System configuration

Before starting to use the FIERA CCD sw WS and SLCU parts, one has to configure in the OLDB the table of SLCU attributed which have to be updated on WS through the CCS Scan System.
To do this, the FIERA CCD sw provides a UNIX script, called *fcdDcsScan.sh* (see also manual page in 4.3). Type from the UNIX shell:

```
> fcdDcsScan.sh $CCDNAME $RTAPENV $CCDLENV
```

If the stand-alone panel is intended to be used, also the attributes needed by its Graphical User Interface must be included in the scan table. In this case, the script *fcdosScan.sh* must be run in addition to the previous one (see also manual page in 4.3):

```
> fcdosScan.sh $CCDNAME $RTAPENV $CCDLENV
```

Note that, in order them to succeed, both WS and SLCU environments must be properly configured and active.

5.7 Configuring the FIERA SLCU

If the SPARC has to be reconfigured, it can be accessed by attaching a terminal to the serial port marked "`ttya`"before booting. When the machine is booted it will allow logins on this tty.

5.7.1 Changing Basic TCP /IP Configuration

This section describes how to change some of the basic TCP/IP parameters this will be necessary when an SLCU is being moved from one subnet to another, or when the SLCU's host name is being changed.

When changing any of the TCP/IP configuration parameters, the operation must be done as a user with "`root`"privileges, either " `root`"or " `fcdrun`".

The TCP/IP configurations which may need to be changed are:

1. SLCU host name
2. SLCU IP address
3. xntp time server

5.7.1.1 SLCU Ethernet Host Name[1]

To change the SPARC's hostname (e.g. from `oldfiera` to `newfiera`) perform the following steps:

1. Edit the file `/etc/hosts`, which should look like:
   ```
   # Internet host table
   #
       127.0.0.1       localhost
       WWW.XXX.YYY.ZZZ oldfiera    loghost
   ```
   Replace `oldfiera` by `newfiera`.
2. Edit the file `/etc/nodename`, which should look like:
   `oldfiera`
   Replace `oldfiera` by `newfiera.`
3. Edit the file `/etc/hostname.le0`, which should look like:
   `oldfiera`
   Replace `oldfiera` by `newfiera.`
4. If existing, edit the file `/etc/motd`, replacing `oldfiera` by `newfiera.`
5. Set the environment:
   ```
   mv \
   $PECS_ROOTDIR/releases/$PECS_RELEASE/etc/locality/misc-
   <oldfiera>.env \
   $PECS_ROOTDIR/releases/$PECS_RELEASE/etc/locality/misc-
   <newfiera>.env
   ```
   and check the environment variables defined in the file (in particular `RTAPENV` could be different, since it is usually related to the hostname)[2]

5.7.1.2 SLCU IP Address

To change a SPARC's IP address perform the following steps:

1. Edit the file `/etc/hosts`, which should look like:
   ```
   # Internet host table
   #
       127.0.0.1       localhost
       WWW.XXX.YYY.ZZZ myfiera   loghost
   ```
   Replace the IP address `WWW.XXX.YYY.ZZZ` by the new IP address for the machine.

2. If the subnet has changed, edit the file `/etc/defaultrouter`, which should look like:
   `WWW.XXX.YYY.254`
   Ensure that the file contains the IP address of the router for this subnet.

1. This paragraph describes the modification of the name of the Ethernet interface. To modify the name of the ATM interface, refer to [8]
2. for older systems not using pecs, e.g. the WFI, the file to be edited is `/vlt/System/config/$HOST.cshrc`

3. If the net has changed, edit the file `/etc/netmasks`, which should look like:
   `WWW.XXX.0.0 255.255.255.0`
   Replace the IP mask `WWW.XXX.0.0` by the new IP mask for the machine.

5.7.1.3 xntp Time Server

The SLCU runs the xntp daemon to keep its system clock synchronized with another time server on the network. If needed, the file `/etc/inet/ntp.conf` must be updated with the IP address of the correct time server

```
server          WWW.XXX.YYY.ZZZ  version 3 prefer
```

In the example above, replace `WWW.XXX.YYY.ZZZ` with the IP address of the time server.

To set the Time Zone (`TZ`) edit the file `/etc/TIMEZONE` defining `TZ` as follows:

```
TZ=UTC
```

If the xntp daemon has not been installed on the machine, refer to [7] for installation instructions.

5.7.2 Environment Variables

Some environment variables are used on the SLCU by the FIERA CCD sw and therefore must be defined in `$PECS_ROOTDIR/releases/$PECS_RELEASE/etc/locality/misc-$HOST.env`[1].
Some of them, **marked with (*), are optional**: if they are not defined, the specified default value is taken.

RTAPENV
name of the SLCU environment where CCD sw runs. Example:

```
> setenv RTAPENV wmyfiera
```

CCDWENV
name of the workstation CCS environment where CCD sw runs. Example:

```
> setenv RTAPENV wmyws
```

INS_ROOT
name of the root directory where data dictionaries, setup and configuration files are stored/retrieved (see [3] for the full directory structure). This should be set to INS_ROOT mounted via NFS from the Instrument Workstation, example:

```
> setenv INS_ROOT /net/<ins_host>/diskc/myws/insroot
```

where `<ins_host>` is the name of the Instrument Workstation.

CCDNAME
name of the CCD camera (**max. 7 characters**). Examples:

```
> setenv CCDNAME myccd
```

INS_USER (*)
INS_ROOT branch for setup files search (see [3] for more details). If not defined, default is `SYSTEM`. Example:

```
> setenv INS_USER SYSTEM
```

1. for older systems not using pecs, e.g. the WFI, the file to be edited is `/vlt/System/config/$HOST.cshrc`

CCDDID (*)
Name of Common Data Interface Dictionary to be used (excluded prefix ESO-VLT-DIC.). If not defined, default is CCDDCS. Example:

```
> setenv CCDDID MYCCD
```

The Data Interface Dictionary file must be stored in $INS_ROOT/SYSTEM/Dictionary.

The default Dictionary file (ESO-VLT-DIC.CCDDCS) is in $VLTROOT/config.

FCDDID (*)
Name of FIERA Data Interface Dictionary to be used (excluded prefix `ESO-VLT-DIC.`). If not defined, default is `FCDDCS`. Example:

```
> setenv FCDDID MYFIERA
```

The Data Interface Dictionary file must be stored in `$INS_ROOT/SYSTEM/Dictionary`.

The default Dictionary file (`ESO-VLT-DIC.FCDDCS`) is in `$VLTROOT/config`.

5.7.3 Configuring the "`fcdrun`" user

The FIERA Software on the Instrument Workstation uses a special user account for executing the CCD S/W on the SPARC.

The `~fcdrun/.rhosts` file allows the Instrument Workstation to execute the FIERA S/W. Check that it contains the line:

```
<InsWs> +
```

```
NOTE: Replace <InsWs> with the Instrument Workstation name.
```

5.7.4 Adding the "Instrument User"

In order for the Online Database (OLDB) to function correctly, the same user id must be used on the Instrument Workstation and the FIERA SLCU for starting and stopping the OLDB environment. To create the Instrument User on the FIERA SLCU run the `fcdAddUser` program (see 6.1).

5.7.4.1 Adding the Instrument User manually

For old version of the FIERASW not containing the `fcdAddUser` program, perform the following steps while logged in as the user "`root`". In the example below, replace "`IUser`" by the Instrument User name.

1. Edit the file `/etc/passwd` adding a line like:

   ```
   IUser:x:<IUser_id>:<IUser_grp>::/export/home/IUser:/bin/bash
   ```

   NOTE: Replace `<IUser_id>` and `<IUser_grp>` with the user id and group for the Instrument User from the Instrument Workstation `/etc/passwd` file.
2. Edit the file `/etc/shadow` adding the line:

   ```
   IUser::11022::::::
   ```
3. Create the home directory for the user:

   ```
   > mkdir ~IUser
   ```
4. Change ownership of the files of the "Instrument User":

```
> chown -R IUser:<IUser_grp> ~IUser
```

5. Populate the home directory of the “ Instrument User”user [1]:

```
> su - IUser
> /etc/pecs/bin/pecssh mklinks -i
```

Reply:

```
PECS_ROOTDIR [/etc/pecs]: “return”
PECS_RELEASE [000]: “return”
[...]
Do you wish to install VUE support files? [y]: “n”
```

```
> exit
su - IUser
> zcat $VLTROOT/templates/forFCD/FieraTemplateUser.tar.gz | tar xf -
> chmod +x ~/bin/*
```

5.7.5 SLCU Environment

To re/configure the FIERA SLCU Environment perform the following steps:

1. Check that the environment is known to the ACC database

   lookup the Online Database environment host

   ```
   > fcdEnvToHost wmyfiera
   ```

   A reply with the FIERA SLCU host name should appear:

   ```
   myfiera
   ```

   If the utility does not answer with the name of the SLCU host, there is a problem with the ACC Database. In this case check that the value of the environment `ACC_HOST` is pointing at the host with the correct ACC database. If this value is incorrect check the setting in the file `$PECS_ROOTDIR/releases/$PECS_RELEASE/etc/locality/apps-all.env`[2].

2. If needed, as “`root`” modify the file `/etc/services`, adding the (local) OLDB environment of the Instrument LCSU and the (remote) OLDB environment of the Instrument Workstation. Do that adding the following lines:

   ```
   <RTAPENV>     <PORT1>/tcp
   <InsWsEnv>    <PORT2>/tcp
   ```

   Substitute `<RTAPENV>` with the name of the local environment, `<InsWsEnv>` with the name of the remote Instrument Workstation environment, `<PORT1>` and `<PORT2>` with the numbers of the ports used for the tcp connections (ask the System Administrator for them)

3. As “`vltmgr`”:

   Edit `$VLTDATA/config/CcsEnvList` adding the following lines:

   ```
   <RTAPENV>  /export/home/vltdata/ENVIRONMENTS/<RTAPENV>
   <CCDWENV> <InsWs>
   ```

---

1. for older systems not using pecs, e.g. the WFI, skip this step and as `InsUser` run instead “`cp ~fcdrun/.??* ~/`”
2. for older systems not using pecs, e.g. the WFI, the file to be edited is `/vlt/System/config/$HOST.cshrc`

Substitute <RTAPENV> with the name of the local environment, <CCDWENV> with the name of the remote Instrument Workstation environment, <InsWs> with the name of the remote Instrument Workstation

4. As "vltmgr"r un:

```
> export CCDWENV=<InsWsEnv>
> fcdinsInstall
```

Substitute <InsWsEnv> with the name of the remote Instrument Workstation environment.

5.7.5.1 SLCU Environment Verification

Verify that the environment has been generated:

```
> dbRead "@${RTAPENV}:<alias>${CCDNAME}.date"
```

The following reply should appear:

```
BYTES value =
```

5.7.6 SLCU Logging System Configuration

To configure the SLCU to log messages onto the Instrument Workstation, edit the file /etc/syslog.conf while logged in as the user "root".

```
# ========================================================
# The following three lines configure the VLT logging system
# ========================================================
#*info;mail,local1,local2.none          /var/adm/messages
#local1.warning                         /vltdata/tmp/logFile
#local2.warning                         /vltdata/tmp/logAuto
*.info;mail,local1,local2.none          @myws
local1.warning                          @myws
local2.warning                          @myws
```

Substitute the Instrument Workstation hostname for "myws". This change will take affect after rebooting the SLCU. ***IMPORTANT***: use tabs for spacing !

NOTE: if the log messages on the Instrument Workstation have a worng format, check in $VLTDATA/ENVIRONMENTS/$RTAPENV/CcsEnvTable that the logManager is started with the option "-f <InsWsEnv>", where <InsWsEnv> is the Instrument Workstation environment.

5.7.7 Rebooting and Halting the SLCU

1. From the console or xterm.

   The FIERA SPARC can be halted by logging in using the user name "halt" and can be rebooted by logging in as "reboot"!

   ```
   login: halt
   ```

   or

   ```
   login: reboot
   ```

2. From the Instrument Workstation.

   To Halt the SPARC from the Instrument Workstation use the following command (see 4.3.10):

   ```
   > fcdDcsSlcuHalt.sh $CCDLENV
   ```

   To Reboot the SPARC from the Instrument Workstation use the following command (see 4.3.11):

   ```
   > fcdDcsSlcuReboot.sh $CCDLENV
   ```

5.8 CCD Camera Installation Verification

5.8.1 Checking the Connectivity between the SLCU and the DSP board

To check that the DSP board is accessible form the SLCU, login the SLCU as "`fcdrun`"and run

```
> fcdHalloDsp
```

The following message should appear:

```
Your FIERA DSP says: Hallo World !
```

NOTE: If this messagge does not appear, check if the DSP driver has been properly installed (see chapter 2.1.1.2 "Installation of drivers for the FIERA SLCU", in [19])

5.8.2 Checking the Connectivity between the WS and the SLCU manually

This section describes how to check manually that the TCP/IP configuration is correct, the OLDB environments are correct and that the ACC database is functioning correctly.

5.8.2.1 Check TCP/IP Connectivity

To check that the SLCU is accessible via the network from the Instrument workstation:

```
> /usr/sbin/ping 'fcdEnvToHost $CCDLENV'
PING odta2.hq.eso.org: 64 byte packets
64 bytes from 134.171.24.38: icmp_seq=0. time=1. ms
64 bytes from 134.171.24.38: icmp_seq=1. time=2. ms
64 bytes from 134.171.24.38: icmp_seq=2. time=2. ms
```

You will need to terminate this using "`^C`".

If this does not succeed there is a problem with the basic TCP/IP configuration. Check the IP address of the SLCU in the `/etc/hosts` file.

5.8.2.2 Check that the Instrument Workstation has permission to execute the S/W

To check that the `fcdrun` user allows the Instrument Workstation to execute the S/W type the following:

> remsh `fcdEnvToHost $CCDLENV` -l fcdrun ls

You should see a listing of the `fcdrun` users home directory. If instead you see an error message

Permission Denied

this implies that the `~fcdrun/.rhosts` file does not have an entry for the Instrument Workstation (see 5.7.3).

5.8.2.3 Check the SPARC Environment is known to the WS

The Workstation must be able to identify the host name on which the SLCU Environment is running. Type the following command

> fcdEnvToHost $CCDLENV

The command should echo the hostname of the SPARC. If it does not, this implies that there is a problem in the ACC Database. Check the value of the environment variable `ACC_HOST`.

5.8.2.4 Check CCS Message System Connectivity

Send a message to the command manager in the SLCU Environment:

> msgSend wmyfiera cmdManager PING " " 10000
> MESSAGEBUFFER:
> OK

If this fails check the following:

1. Does the `$CCDLENV` environment have an entry in `/etc/services` on the Instrument Workstation which matches the entry in `/etc/services` on the SLCU.
2. Does the SLCU have an entry in `/etc/services` for the $RTAPENV which matches the entry in `/etc/services` on the Instrument Workstation.
3. Is the same user account being used on the Instrument workstation and the SLCU to start the environments ? If the same user account with the same user id is not used on both machines, communication will not work.

5.8.2.5 Checking the `INS_ROOT` Mounting on the SLCU

To check that the SLCU is correctly mounting the `INS_ROOT` from the Instrument Workstation run:

> remsh `fcdEnvToHost $CCDLENV` -l fcdrun 'echo $INS_ROOT'

a reply looking like the following should appear, where `<ins_host>` is the name of the Instrument Workstation:

`/net/<ins_host>/...`

then run:

> remsh `fcdEnvToHost $CCDLENV` -l fcdrun 'ls $INS_ROOT'

the following reply should appear:

```
SYSTEM
```

If this fails, the Instrument Workstation is probably not exporting the disks to the SLCU. Check the `/etc/exports` file on the Instrument Workstation: there should be a line like:

/diska -anon=65534,access=oldfiera

Replace all the occurrences of `oldfiera` by `newfiera`, and run the command

> exportfs -a

5.9 CCD Software Operation Verification

To get familiar with the user interface execute the " Simple Demo Session"as described in 8.4.
To run the WS processes execute the " SLCU Simulated Session"as described in 8.4.

5.10 Using the FIERA CCD Software

After successful completion of all the steps described in this chapter, the FIERA CCD sw and the environment are ready for usage.
It is suggested to follow the steps described in 8.8.

6 REFERENCE FOR CHECKING OF THE FIERA CONFIGURATION

The programs available for the checking of the FIERA configuration are:

1. **fcdAddUser**. Install an Instrument User on the FIERA SPARC.

6.1 fcdAddUser

NAME

```
fcdAddUser - add a user to a SLCU system and/or configures dot files
```

SYNOPSIS

```
fcdAddUser -n user_name [-u uid] [-g gid] -e RTAPENV -c CCDNAME
           [-int INTROOT] -ins INS_ROOT [-d] [-conf]
```

DESCRIPTION

```
This command must be run as root. It allows the creation of a user
and/or the configuration of dot files using in astandard way.
If the user name is "fcdrun" the user id defaults to "0" and any other
value given by the user is ignored. If the home directory of the new
user already exists, the dot files are not updated. It is necessary
to run again the script with "-conf" switch. Users are configured
without password. To give them a password use passwd(1).
<-n user_name>  name of the user to add to the system
<-u uid>        uid of the user to add to the system (optional)
<-g gid>        gid of the user to add to the system (optional)
<-d>            permits duplication of the user id (optional)
<-conf>         configure dot files with given RTAPENV, INTROOT,
                CCDNAME, INS_ROOT, INTROOT. The user must already exist
                (optional)
<-e RTAPENV>    RTAP environment
<-int INTROOT>  INTROOT directory
<-ins INS_ROOT> INS_ROOT directory
<-c CCDNAME>    camera name
```

```
- - - - - -
Last change:  01/04/04-08:11
```

7 ERROR DEFINITIONS

The FIERA CCD software uses the standard mechanism defined and provided by the CCS error system to log and return error information to external applications, both at function and command level.
Errors are defined for both WS and SLCU in the following files:

1. **ccdErrors.h** error codes
2. **ccd_ERRORS** error messages

See sections 4.5 and 4.10 for more information.

8 STANDALONE OPERATIONS

This part of the document provides a description of the way how a **CCD standalone** system must be operated. Operations are performed through GUI panels; no programming is involved nor required. On the other hand, the DCS part of the CCD software, does not include any GUI panel and can be commanded, also from a remote station, through messages sent by higher level applications belonging to instrumentation software or telescope control software.

8.1 Configuration

Before being able to use a CCD camera, its configuration must be defined and saved. The same operation must be performed each time some change in the camera configuration takes place (e.g. a new chip or a new cryostat is mounted).

A prerequisite to be able to configure a CCD camera is that the VLT CCD software must already have been installed and configured (see Chapter5).

The configuration of a CCD camera does not require an in-depth knowledge either of the CCD software or of the VLT software in general, in that the CCD software provides panels to help and guide in this operation. On the other hand it requires knowledge of the characteristics of the camera to be configured.

Note: the usage of Configuration panels is restricted to off-line operations. The setting of configuration values acts on the on-line database; it is recommended not to change any configuration value during camera operations: the behavior of the running software could become unpredictable.

Type from the Workstation shell prompt:

```
> fcdConfig &
```

A panel as shown in Fig.5 will appear. For details about the contents see manpage at section 9.1.

The directory where files are saved/loaded from is
`$INS_ROOT/$INS_USER/COMMON/CONFIGFILES`
(or `$INS_ROOT/SYSTEM/COMMON/CONFIGFILES` if `INS_USER` is not defined).
(see Chapter5 for the meaning of the environment variable `INS_ROOT` and `INS_USER`)

To configure usage of “ image extensions per port”select the relevant option.

To configure readout modes press the *Modes Configuration* button, the *fcdConfigMode* panel (see Fig.6) is then called. For details about the contents see the fcdConfigMode manpage at section 9.1.6.

To configure telemetry (temperature and pressure sensors) press the *Telemetry Configuration* button, the *fcdConfigTel* panel (see Fig.6) is then called. For details about the contents see the fcdConfigTel manpage at section 9.1.6.

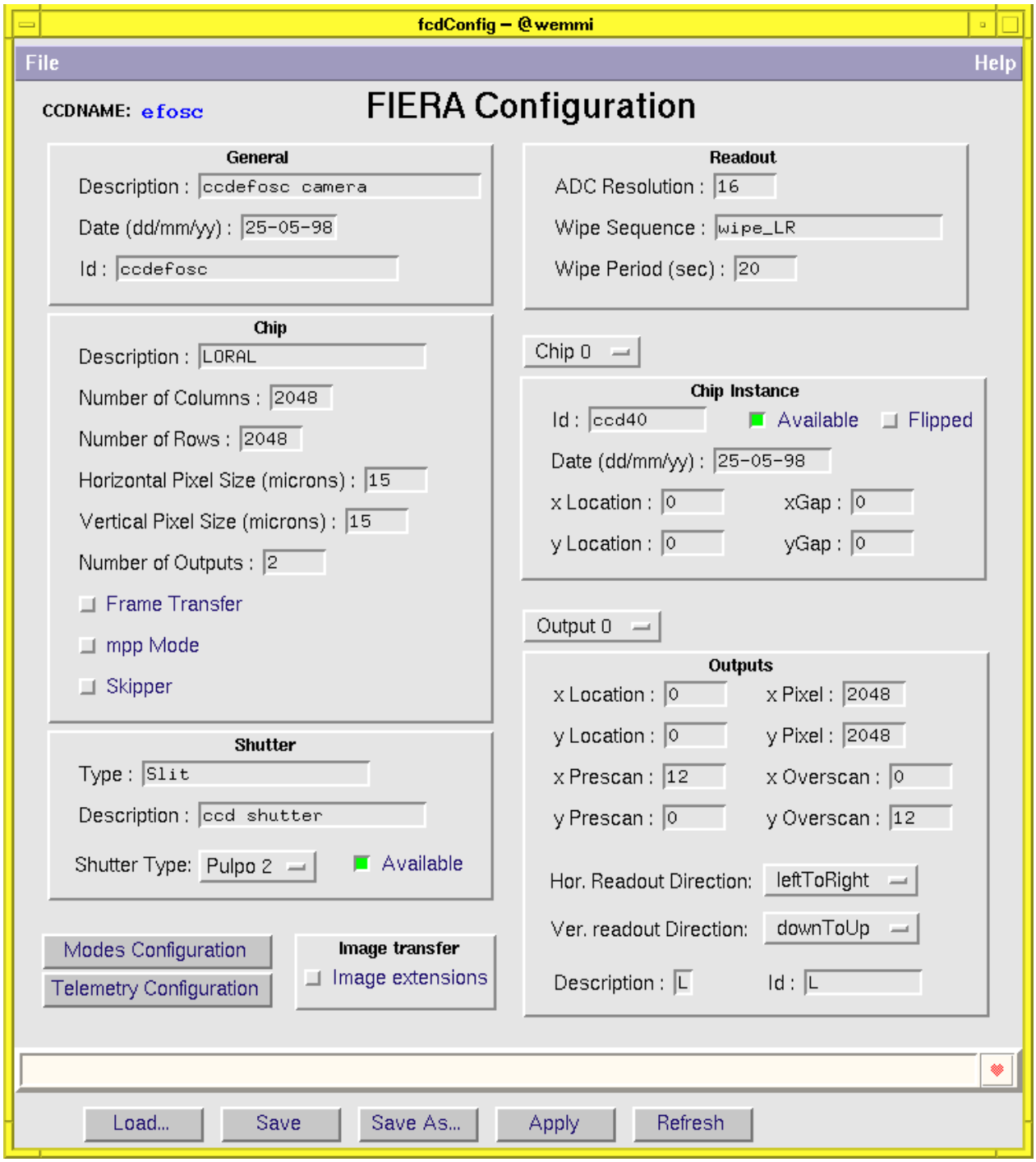


Fig.5 CCD camera configuration panel

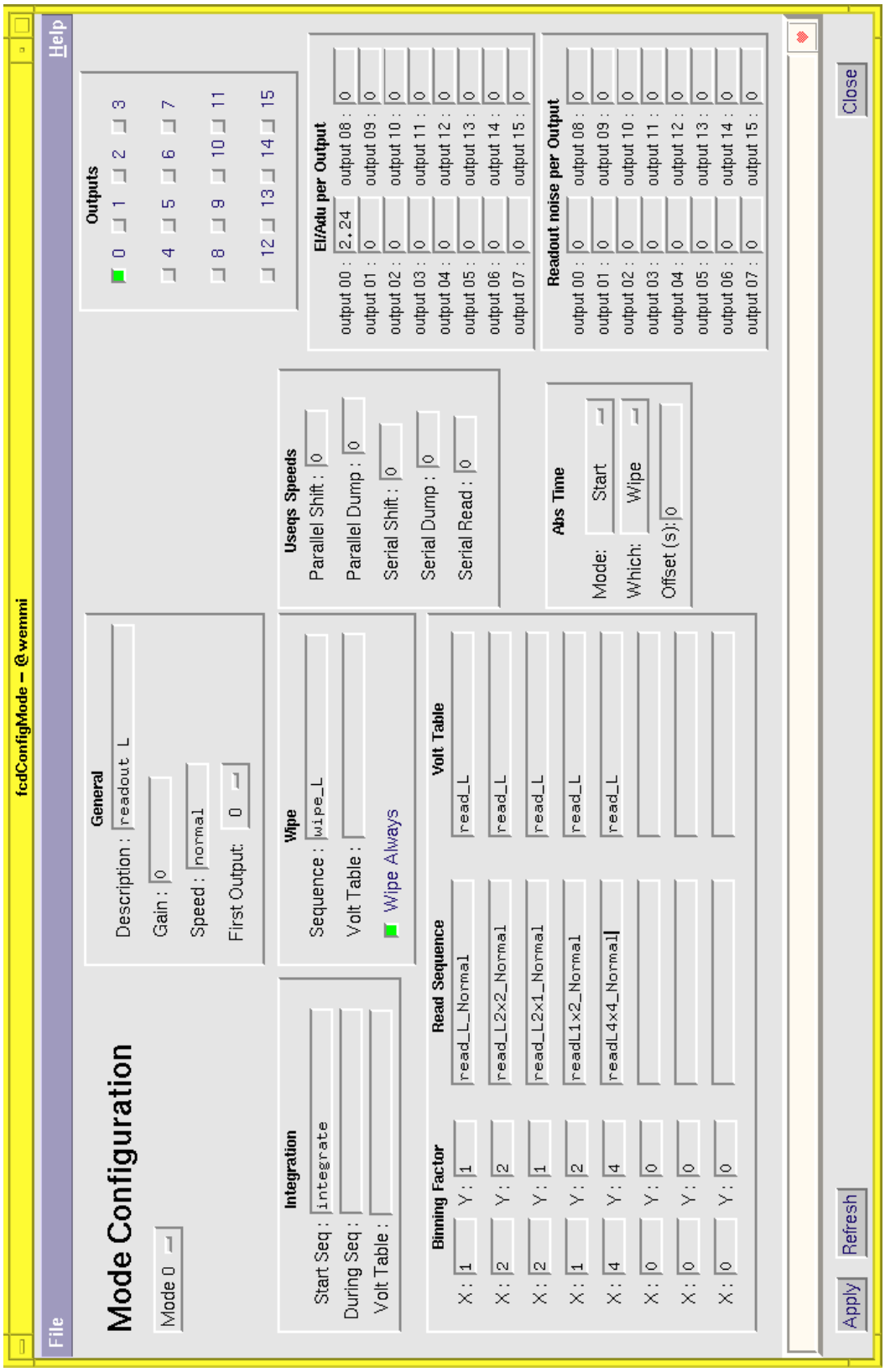


Fig.6 CCD Modes configuration panel

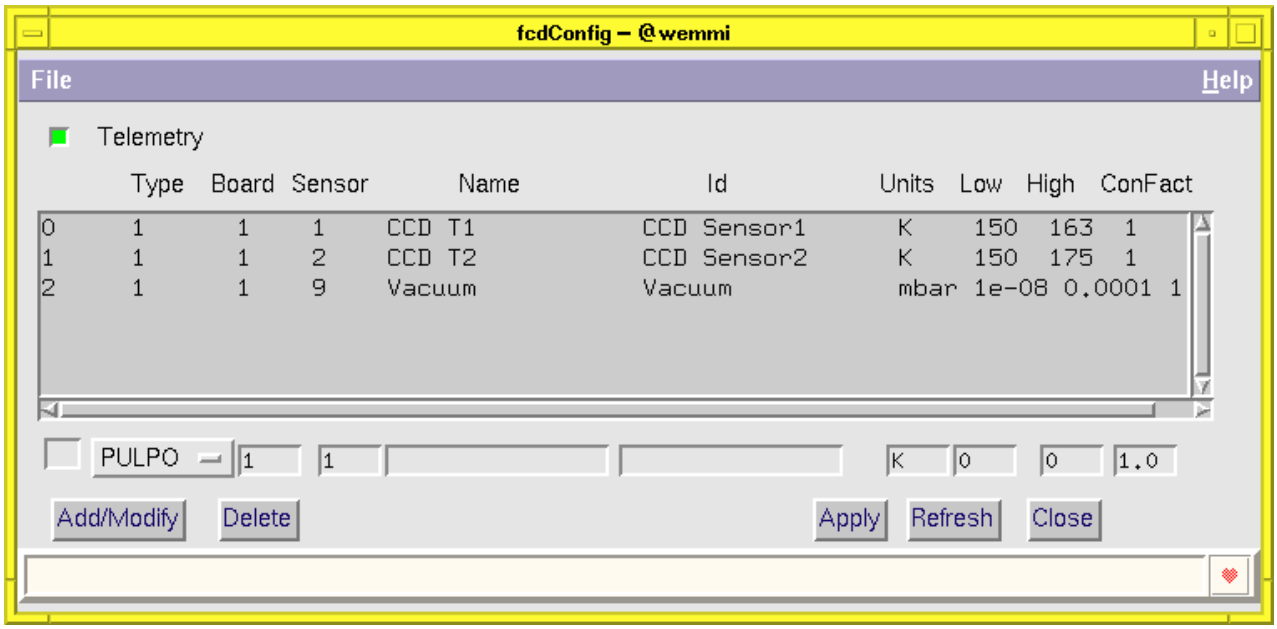


Fig.7 Telemetry Configuration Panel

8.2 Control panel

One single panel, shown in Fig.8, provides all functionality needed to startup/shutdown the CCD software, define an exposure setup, start and control an exposure, display an image as result of an exposure.
The same panel is used, independently if and which parts of the CCD system used are simulated.
Detailed information about meaning and scope of each widget in the panel can be found in the manual page of *fcdCtrl*, section 9.1
Note: the panel is rather big and contains quite some information. Depending on the load of the Workstation CPU it can take a few seconds before being displayed. Please, wait and don't panic!

8.3 Engineering Interface

A GUI panel is provided to help engineers in case of trouble. It enables the most common operations needed for engineering and trouble-shooting.
The panel gives freedom to do actions at a low level and must be used with care !
It is assumed that the user knows the CCD sw and the VLT sw environment and is fully aware of the actions associated to each button and possible consequences.

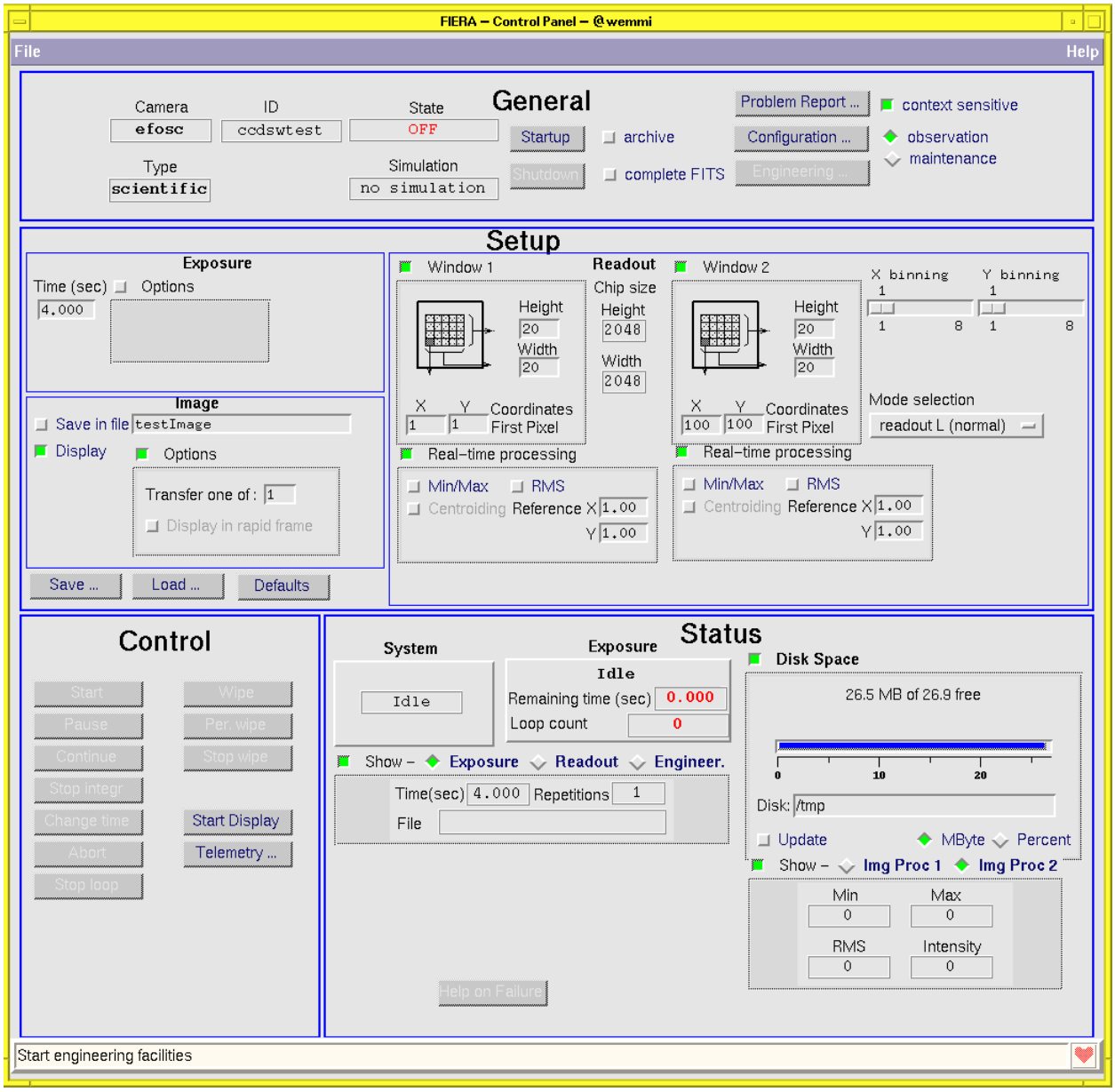


Fig.8 FIERA CCD Control panel

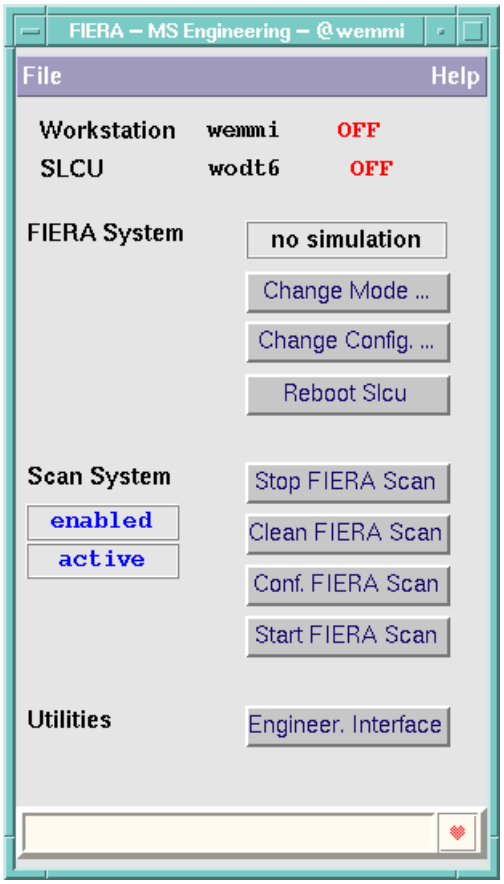


Fig.9 FIERA CCD engineering panel

To start the " FIERA Engineering Panel"you can either

1. Enable the "*maintenance*" radio button in the "*General*" area of the *FIERA Control Panel*, and then push the "*Engineering ...*" button

or

2. Type from the Workstation shell prompt:

```
> fcdei &
```

A panel as shown in Fig.9 will appear. For more details, see section 9.1

8.4 Getting started

In order to be able to operate the CCD software, the installation procedure, as described in Chapter 5 must have been completed successfully.
As already mentioned, the CCD software runs on different platforms. In order to help beginners in the learning phase and to localize better possible cause of failure, we suggest getting into the system gradually, executing in sequence:

1. A simple demo session (the whole CCD sw is simulated within the control panel).
2. A brief session with SLCU simulated (the complete SLCU sw is simulated, see also section 8.6)

Note: In the following, the usage of the mouse is limited to the left button, e.g. to push buttons and select entry fields in panels.

8.4.1 Simple Demo session

Type from the Workstation shell prompt:

```
> fcdDemo.sh
```

The control panel (Fig.8) is displayed.

Execute in sequence the following operations:

1. Start an observation session (area *General*).
    a. Push the button *Startup*. The CCD state will turn to *on-line*.
2. Retrieve an existing setup file into the panel (area *Setup*).
    a. Push button "*Load ...*". A file selection box appears.
    b. Enter in the Selection field file name *fcdSetupComplete.det*
    c. Push OK. The file selection box disappears and the entries in the *Setup* area are updated according to the contents of the setup file.
3. Change a few setup values.
    a. Area *Exposure*
        i. Enter Time 30
    b. Area *Image:*
        i. Enter file name *fcdImageWsSim.fits*
        ii. Unselect *Display*
4. Save the defined setup in a setup file.
    a. Push button "*Save ...*". A file selection box appears.
    b. Enter in the Selection field file name *myFirstSetup.det*
    c. Push OK. The file selection box disappears and your setup definition is saved
5. Simulate cleaning the chip (area *Control*).
    a. Push the *Per. Wipe* button. The string *Wiping* appears in the *System* box (area *Status*).
    b. Push the *Stop wipe* button. The string *Wiping* disappears.
6. Run one single exposure (area *Control*).
    a. Push the button *Start*.This will animate the area *Status*, simulating the execution of the exposure.
    b. Push the button *Pause.* The status of the running exposure goes to *paused* and the remaining time remains fixed
    c. Push the button *Continue*. The status of the running exposure goes to *integrating* and the remaining time decreases again.
    d. Push the button *Stop integr.* The remaining time goes to 0 and the exposure status changes to *Read&Transfer.* After a while it changes to *Finished*.
7. Run a loop of repeated exposures.
    a. Area *Setup Exposure*
        i. Select *Options*

ii. Select *Repeat forever*

b. Push the button *Start*.The string *Loop active* will appear the area *Exposure Status*.

c. Push the *Stop loop* button. The current exposure is terminated regularly and then no further exposure is started.

d. Push the button *Abort*. The execution of the current exposure is immediately stopped.

8. Monitor telemetry and temperature values

a. Push the button *Telem/Temp...* (area *Control*). The panel dedicated to telemetry and temperature monitoring is displayed.

b. Push the button *Start* in the *Telemetry* area. The telemetry status will turn to *Active* and the values displayed are updated periodically.

c. Select any telemetry parameter in the *Telemetry Plot* area, button labelled 1.

d. Push the button *Start* in the *Telemetry Plot* area. A new window pops-up, showing the plotting of the selected parameter.

e. Push the button *Stop* in the *Telemetry Plot* area. The plot disappears.

f. Push the button *Stop* in the *Telemetry* area. The telemetry status will turn to *Idle*.

g. Push the button *Start* in the *Temperature* area. The temperature status will turn to *Active* and the values displayed are updated periodically.

h. Push the button *Stop* in the *Temperature* area. The temperature status will turn to *Idle*.

i. Close the panel (select menu item *File* option *Quit*).

9. Terminate the session.

a. Push the button *Shutdown* (area *General*). The CCD state will turn to *OFF*.

b. Close the panel (select menu item *File* option *Quit*).

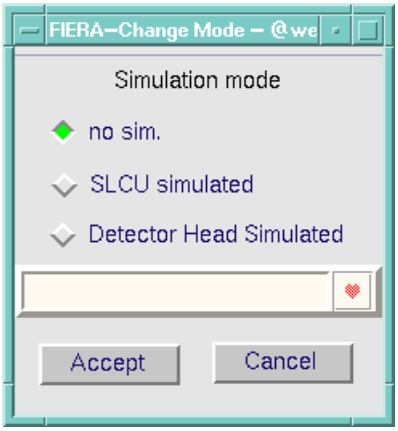


Fig.10 Change Operational Mode panel

8.4.2 SLCU simulated session

Type from the Workstation shell prompt:

```
> fcdStart.sh
```

The control panel (Fig.8) is displayed.

Execute in sequence the following operations:

1. Check *State* in the area *General*. If it is OFF, go to 2., otherwise:
   a. Check the status of the button *Shutdown*: if it is disabled, unselect option *context sensitive.* The button *Shutdown* is enabled
   b. Push the button *Shutdown*. After a while the *State* will turn to *OFF*.
2. Set the operational mode
   a. Select the maintenance option (area *General*). The button *Engineering* is enabled
   b. Push the button *Engineering*. The panel *Engineering* pops up (see 8.3).
   c. Push the button *Change Mode (*area *CCD System)*. The panel *Change mode* (Fig.10) pops up
   d. Select option *SLCU simulated*
   e. Push *Accept.* The panel disappears and the simulation field in the area *General* of the main panel turns to SLCU *simulated*
   f. Close the panel *Engineering* (select menu item *File* option *Quit*).
3. Start an observation session simulating the whole SLCU sw.
   a. Push the button *Startup* (area *General*). After a while the CCD state will turn to *on-line*.
4. Retrieve an existing setup file into the panel (area *Setup*).
   a. Push button "*Load ...*" A file selection box appears.
   b. Enter in the Selection field file name *fcdSetupComplete.det*
   c. Push OK. The file selection box disappears and the entries in the *Setup* area are updated according to the contents of the setup file.
5. Start Real-time display (area *Control*).
   a. Verify if *rtdServer* is running:
      ```
      > ps -aef | grep rtdServer | grep -v grep
      ```
   b. If it is not running, start it:
      ```
      > rtdServer &
      ```
   c. Push the *Start Display* button. It takes a while to pop up the *Real-Time Image Display* utility.
   d. In the latter panel, select the menu entry *Real-time,* item *Attach Camera.*
6. Modify the setup (area *Setup*).
   a. Area *Exposure*
      i. Enter Time 3
   b. Area *Image:*
      i. Enter file name *fcdImageLcuSim.fits*
      ii. Select *Display* (only if no problems encountered at point 5. above, otherwise unselect *Display*)
7. Simulate cleaning the chip (area *Control*).
   a. Push the *Per. Wipe* button. The string *Wiping* appears in the *System* box (area *Status*).
   b. Push the *Stop wipe* button. The string *Wiping* disappears.
8. Run an exposure (area *Control*).

a. Push the button *Start*.This will animate the area *Status*, simulating the execution of the exposure. When SLCU is simulated, all the fields are updated but the Readout speed in the Status area. As a result of the simulated exposure, an artificial image, consisting of a linear scale (1,2,3....) is saved in the specified file in FITS format and displayed with the Real-Time Image Display utility, the latter only if real-time display has been selected in the setup definition (point 6.b.ii.); to improve the quality of the displayed image press the button *Auto Set Cut Levels*

9. Terminate the session.
   a. Close the Real-Time Display panel (select menu item *File* option *Quit*)
   b. Push the button *Shutdown* (area *General*). The *State* will turn to *OFF*.
   c. Close the main panel (select menu item *File* option *Quit*).

8.5 Exposure definitions

8.5.1 Exposure types

The CCD standalone operation supports only exposure types as known to FIERA CCD DCS (see [5] and 2.3.2).

8.6 Operational modes and simulation

The operational modes implemented are:

1. **Normal**. The FIERA CCD software tries to access the related hardware, assuming that the whole software and hardware exist and are properly installed.
2. **Simulation**, at various levels:
   a. **SLCU software simulated.** The whole SLCU SW is simulated at WS level. The SLCU processes are simulated by means of the simulation option in the CCS Message System. This kind of simulation is useful when no SLCU is available; the user must be aware that at this level of simulation **the behaviour of the CCD camera is rather simplified**. Nevertheless the interface towards the external software can be almost completely tested as for *Normal* mode.
   b. **Hardware simulated.** The SLCU software behaves the same as in *Normal* mode, except that the Detector Electronics are not accessed. The SLCU assumes that no failure takes place and the most realistic results are returned. This mechanism is implemented in a way that the hardware does not need to exist at all and even the device drivers do not need to be installed.

Changing the operational mode is allowed ONLY when the system state is OFF (see 8.7). The desired operational mode has to be configured in the local database (both in the WS and SLCU!) before start-up (see panel *CCD Change Mode* Fig.10 and section 8.8.3).

8.7 Operational states

The CCD software can be in one of the following operational states (see [3] for standard states definition).

1. **OFF.** The CCD software is in OFF state when it is not loaded or some task is not running. No operations are possible in this state.
2. **LOADED.** The CCD software goes to LOADED state as soon the database is loaded and all processes needed after cold start-up are activated.
3. **STAND-BY.** Normally all hardware components are powered-off, except those which are needed to preserve the CCD from damages, such as the temperature control and the periodical chip wiping.

   In detail all actions needed to bring the whole CCD camera to standby mode are very dependent on the system hardware architecture and therefore cannot be defined in this document for all cameras. Typically the following actions are implemented:
   a. The output relays to the CCD are open (CCD disconnected).
   b. Shutter control hardware is switched off, whenever the hardware architecture allows it.
   c. Temperature control remains active (*not in the present release*)
   d. LAN connection active (command reception enabled)
4. **ON-LINE**. This is the only state where the CCD software can perform exposures.

8.8 Running the system

In order to be able to operate the FIERA CCD software, **the installation procedure,** as described in Chapter 5, has to be followed and completed successfully.

8.8.1 Starting the control panel

Type from the Workstation shell prompt:

```
> fcdStart.sh
```

The control panel (Fig.8) is displayed.

8.8.2 Checking CCD camera configuration

Before starting to operate the camera it is essential to verify that the configuration parameters stored in the on-line database are properly set.
Push the button "*Configuration ..*" in the area *General.* The CCD configuration panel, described in 8.1, pops-up.
In order to verify the correctness of the parameters stored, knowledge of the CCD camera characteristics is needed, and therefore assistance of a specialist, at least for the very first verification is recommended.

8.8.3 Setting the operational mode

1. Check *State* in the area *General*. If it is OFF, go to 2., otherwise:
   a. Check the status of the button *Shutdown*: if it is disabled, unselect option *context sensitive.* The button *Shutdown* is enabled
   b. Push the button *Shutdown*. After a while the *State* will turn to *OFF*.
2. Set the operational mode

a. Select the *maintenance* option (area *General*). The button *Engineering* is enabled
b. Push the button *Engineering*. The panel *Engineering* pops up (see 8.3).
c. Push the button *Change Mode (*area *CCD System).* The panel *Change mode* (Fig.10) pops up
d. Select option *no sim.*
e. Push *Accept.* The panel disappears and the simulation field in the area *General* of the main panel turns to *no simulation* (or the desired simulation mode if this is the case)
f. Close the panel *Engineering* as well (select menu item *File* option *Quit*).

3. Define the contents of the header in the FITS files produced:
   a. All information (basic and HIERARCH FITS) in the image file.
      Select *complete FITS* option (area *General*).
   b. basic FITS information in the image file and HIERARCH information in a separate ASCII file with extension *.det* (default, same output produced by the CCD DCS part).
      Unselect *complete FITS* option (area *General*).
4. Define, through the option *Archive* (area *General*), if the VLT Archive System has to be informed or not about new files produced by the CCD standalone sw. The interface is defined in [6].

8.8.4 Starting the control software

Push the button *Startup* (area *General*). After a while the CCD state will turn to *on-line*.

8.8.5 Defining the exposure setups

The whole Setup area of the control panel is dedicated to this purpose. See section 9.1 for more information about its contents.
The user can:

1. Define the setup for the next exposure to be started.
2. Save the displayed setup in a setup file for later usage (button *Save*).
3. Retrieve a previously defined setup from a setup file (button *Load*)

Note 1: The current version of the CCD software does not allow changes to the setup parameters while an exposure is running. It is therefore recommended to check carefully the setup definition before starting an exposure.

8.8.6 Controlling the exposures

This is accomplished through the buttons in the *Control* area.
The status of a running exposure is displayed in the *Status* area.

It is suggested, before starting any exposure, but especially the very first one, to clean the chip from charges possibly accumulated during periods when the camera was not active. To do this, push the button *Start wipe.* One complete clean-up cycle of the whole chip may take, depending on the chip size, a few seconds. Push *Stop wipe* to terminate the wiping.

To start an exposure, push the button *Start.* The setup currently displayed will be first passed to the system (for the very first exposure it takes a few seconds longer, since the whole Data Dictionary has to be scanned) and then the exposure is started.This will animate the *Status* area.

The actions which can be performed while an exposure is running are:

1. Pause the exposure. Button *Pause*.
2. Continue a paused exposure. Button *Continue*.
3. Stop the current integration and start immediately the read-out. Button *Stop integr.*
4. Abort an exposure. Button *Abort.* The current version of the CCD software only allows an exposure to be aborted while it is integrating. It is not possible to interrupt the read-out.
5. Stop a loop of repeated exposures. Button *Stop loop*. Note that the exposure currently running will be terminated regularly. Therefore, if one is not interested in it either, the *Abort* button must be pushed additionally.

8.8.7 Failure reporting

Failures can be divided in the following categories:

1. *Synchronous*. They occur while executing a command. They are normally reported to the user automatically through a pop-up window.
2. *Asynchronous*. They occur while some background activity is running. Typical example is a failure during the execution of an exposure. The exposure itself is started through the command *START*; the execution of the command is completed when the exposure is started; if a failure occurs after the exposure has been started, then there is no automatic mechanism to report to the user why such a failure occurred. Such information can be retrieved by pushing the button *Help on Failure* in the *Status* area of the main panel.

8.8.8 Analysing the images

The image files produced by the CCD sw standalone are in FITS format. They can therefore be analyzed by any image analysis software supporting FITS, e.g. ESO-MIDAS.

Furthermore, for quick-look, one can start the VLT Real-Time Display (Rtd) utility (button *Start Display*). When the *Rtd* new panel appears (it takes a few seconds), in order to display the CCD images during acquisition, select the menu entry *Real-time*, item *Attach Camera.* Of course, the image display option must have been selected during the setup operations (see section 8.8.5).

8.8.9 Monitoring the telemetry and the temperature values

Push the button ***Telemetry*** in the "*Control*" area of the *FIERA Control Panel*. After a while the Telemetry panel shown in Fig.11 will be displayed

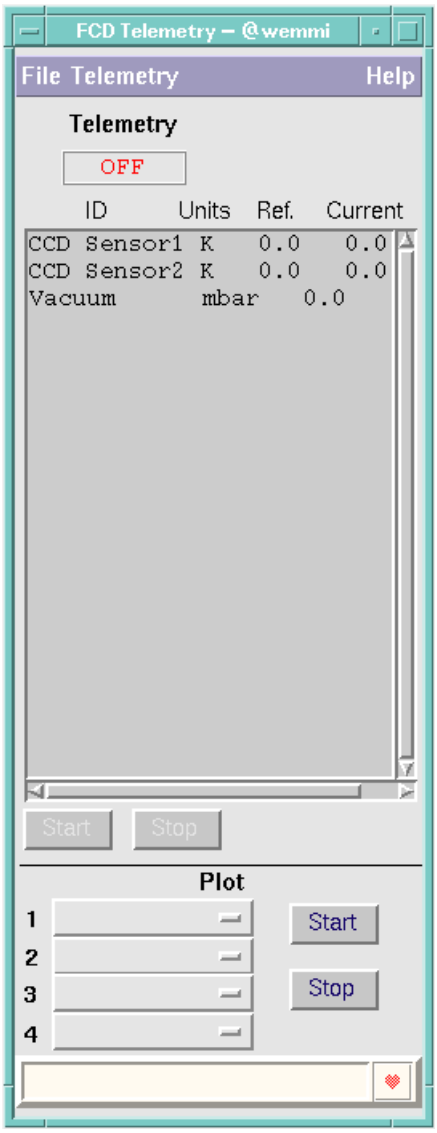


Fig.11 Telemetry Display

This panel allows telemetry monitoring to be started and stopped and values to be plotted using RTAP plotting tools.

8.8.10 Shutdown of the control software

Push the button *Shutdown* (area *General*). The *State* will turn to *OFF*.

8.8.11 Shutdown of the environment

1. Close the Real-Time Display panel, if active (select menu item *File* option *Quit*)
2. Close the main panel (select menu item *File* option *Quit*).

9 \ STANDALONE OPERATIONS: REFERENCE

This chapter provides a detailed description in terms of manual pages of the panels, scripts and the options available for the FIERA CCD stand-alone.

9.1 Panels

In this section the manual pages for the following panels are presented:

1. CCD control panel (*fcdCtrl*),
2. CCD Telemetry Display panel (fcdTelemetry),
3. CCD Change Mode panel (*fcdOpMode*),
4. CCD MS Configuration panel (*fcdConfig*),
5. CCD MS Telemetry Configuration panel (fcdConfigTel),
6. CCD MS Chips Configuration panel (*fcdConfigChipInstance*),
7. CCD MS Outputs Configuration panel (*fcdConfigOutputs*),
8. CCD MS Readout Modes Configuration panel (*fcdConfigMode*),
9. CCD MS Engineering Interface panel (*fcdei*).

9.1.1 fcdCtrl

NAME

```
fcdCtrl - Main panel for FIERA stand-alone operations
```

MENU BAR

```
"File" menu
    Quit
        exit the application

"Help" menu
    Extended Help. Not implemented yet
```

APPLICATION AREA

```
"General"
  "Camera"
    The line below shows the name of the camera according to
    environment variable CCDNAME
  "Type"
    The line below shows if the camera is of type scientific or technical.
    It is derived from the name of the clock patterns.
  "ID"
    The line below shows an unique identifier of the camera used.
  "Chip ID"
    The line below shows an unique identifier of the chip used.
  "State"
    The line below shows the current state of the FIERA sw.
  "Simulation"
    The line below shows the level of simulation the software is supposed
    to run.
  "Startup"
    Perform a complete ("cold") startup of the whole FIERA sw.
    If the software is already partially or fully running, first it is
    shut-down.
    It takes a few seconds.
    After successful completion of the startup, the operational state
    becomes on-line
  "Shutdown"
    Shutdown the FIERA sw. After successful completion, the operational
    state becomes OFF
  "complete FITS"
    If enabled, the HIERARCH FITS information, normally saved in a
    separate ASCII file, is merged into the image FITS file
  "archive"
    If enabled, the FIERA stand-alone sw informs VLT archive about new
    image data available whenever a new image is saved on disk.
  "Problem Report"
    It calls the sprBrowse utility. Enabled only in those environment
    where this utility is available and in the search PATH for executable.
  "Configuration"
    The current values of the configuration parameters, as stored in the
    on-line database, is shown through the panel fcdConfig.
  "Engineering"
    Start the engineering panel fcdei for FIERA operations.
  "Context sensitive"
    If selected, buttons are enabled/disabled according to the current
    context. If not selected, all buttons are enabled, independently
    from the current status of the system.
  "Observation/Maintenance"
    It allows to select the level of privilege (e.g. if in observation,
```

```
    the engineering button is disabled and the configuration parameters
    can be shown, but not modified).
    Normally to enter maintenance mode, one must have a special privilege
    (e.g. password), currently not implemented.
"Setup"
  "Exposure"
    "Time"
      Exposure time in seconds
    "Options"
      "Repeat"
        "forever" --> infinite loop of repeated exposures
        "n times" --> repeat the exposure n times
      "Period"
        Periodicity, in sec, for repeated exposures. If set to 0,
        the next exposure is started as soon the previous one is finished.
        Meaningful only for repeated exposures.
      "Type"
        Currently only options are Normal and Dark (the latter disabled
        if the system has no shutter)
  "Image"
    "Save in file"
      If set, the image is saved in the specified file in
      FITS uncompressed format
    "Display"
      If set, the image is written in shared memory for Real-Time Display.
    "Options"
      "Type"
        Type of image which is going to be produced.
        Options:
          "Normal". Normal case
          "Bias". The image has to be saved as bias frame for processing
          "FF". The image has to be saved as flat field frame for process
      "Transfer"
        Transfer to WS only one image out of n acquired.
        Meaningful only for repeated exposures.
      "Display in rapid frame"
        The image is displayed in the Rtd rapid frame instead of the main
        frame
  "Readout"
    "Window n"
      If set, the readout of that window is enabled.
      If no window is enabled, then the whole frame is read
      "Coordinates first pixel"
        Coordinates of the lower left pixel in the window (first pixel 1,1)
      "Height, Width"
        Window dimension
    "Real-time processing"
      "Min/Max"
        If set, the SLCU computes the minimum and maximum pixel value
        in the whole image
      "RMS"
        If set, the SLCU computes the RMS value in the whole image
      "Centroiding"
        If set, the SLCU computes the centroid for that window
        "Reference"
          Coordinates of the reference pixel for centroiding error vector
    "Binning"
      Binning in both direction applied to the chip readout.
    "Mode"
      Select the wanted readout mode among those supported
"Control"
  "Start"
    Start a new exposure with the setup parameters shown in the Setup area
```

```
    "Pause"
      Pause a running integration
    "Continue"
      Continue integrating after a pause
    "Stop integr."
      Stop immediately the current integration and read-out the chip
    "Abort"
      Abort the current exposure. Image data are lost !!!
    "Stop loop"
      It applies only to repeated (possibly forever) exposures. The current
      exposure is terminated regularly and no new exposure is started.
    "Wipe"
      Wipe the CCD chip once
    "Per. wipe"
      Stop periodical wipe of the CCD chip
    "Stop wipe"
      Start to wipe the CCD chip periodically
    "Start Display"
      Start the rtd demo application as Real_time Display facility
    "Telemetry"
      Start the panel for telemetry and temperature monitoring
  "Status"
    "Help on Failure"
      Information about the last asynchronous error logged by the SLCU sw
      can be retrieved. Enabled only if failure is still actual.
    "System"
      Major information about system status (shutter, readout) is given
      in this box
    "Exposure"
      See manual page of fcdExpStatus_uifClass
    "Disk space"
      See manual page of istDiskMon
    "Show"
      "Exposure"
        See manual page of fcdExpSetup_uifClass
      "Readout"
        See manual page of fcdReadoutSetup_uifClass
      "Engineer."
        Show additional engineering information about on-going exposure
    "Show"
      "Img Proc n"
        See manual page of fcdIpStatus_uifClass
```

SEE ALSO

```
fcdOpMode, fcdConfig, sprBrowse, fcdei
fcdExpStatus_uifClass, fcdIpStatus_uifClass, istDiskMon
fcdExpSetup_uifClass, fcdReadoutSetup_uifClass
```

CAVEATS

```
Some values read from the on-line database have a fixed refresh rate.
The change of some values, although faster in the database, might be
shown on the panel with some delay (up to 2 seconds).
```

```
- - - - - -
Last change:  01/04/04-08:11
```

9.1.2 fcdTelemetry

NAME

fcdTelemetry - panel for CCD telemetry and temperature monitoring

MENU BAR

"File" menu
Quit
exit the application

"Help" menu
Extended Help. Not implemented yet

APPLICATION AREA

"Telemetry"
On the same line the current status of the telemetry process is shown
Below is a table showing information about the current status of
single telemetry values:
"ID"
Unique identifier of telemetry value
"Units"
Units the value is represented
"Ref."
Reference value
"Current"
Current value

The buttons corresponding to possible actions on telemetry values are:
"Start"
Start monitoring telemetry values
"Stop"
Stop monitoring telemetry values

"Plot"
By selecting one of the telemetry values a plot can be made using the
"Start" button, and stopped using the "Stop" button.

SEE ALSO

fcdCtrl

- - - - - -
Last change: 01/04/04-08:11

9.1.3 fcdOpMode

NAME

```
fcdOpMode - Panel used to change the operational mode of the CCD sw
```

APPLICATION AREA

```
"Simulation mode"
    "no sim."
        no simulation, the whole hardware is used.
    "SLCU simulated"
        The whole SLCU software is simulated at workstation level
    "Detector Head Simulated"
        The hardware is simulated at SLCU level
```

ACTION BUTTONS

```
"Accept"
    The currently selected operational mode is set.
    Note: a change of the operational mode is allowed only when the
    CCD sw operational state is OFF.

"Cancel"
    Close this panel without action.
```

SEE ALSO

```
ccsCtrl
```

```
- - - - - -
Last change:  01/04/04-08:11
```

9.1.4 fcdConfig

NAME

```
fcdConfig - panel for FIERA camera configuration
```

MENU BAR

```
"File" menu
    Quit
        exit the application

"Help" menu
    Extended Help. Not implemented yet
```

APPLICATION AREA

```
Note: Entries marked with (*) have influence on the camera operations.
      They must be set to the proper value in order to get the camera
      behaving as expected.
      All other parameters have no influence on the camera behaviour;
      they are used only as information to be put in the image FITS
      header.

On the main panel the camera name, as defined in the environment variable
CCDNAME, is displayed

"General"
    "Description"
         Description of the camera.
    "Date"
         Date of camera installation (format dd/mm/yy)
    "Identifier"
         ESO wide unique identification code of the camera used.

"Shutter"
(*) "Available"
         The button indicates if a shutter is mounted or not.
    "Description"
         Name of the shutter, including type and main characteristics.
    "Identifier"
         ESO wide unique identification code of the shutter used.
    "Type"
(*)      Type of the shutter i.e. Pulpo (1 or 2), digital or external.

"Chip"
    "Description"
         Name (containing characteristics) of the chip used.
(*) "Number of Columns"
         Number of columns (X).
(*) "Number of Rows"
         Number of rows (Y).
    "Horizontal Pixel Size"
         Horizontal size of each pixel (in microns).
    "Vertical Pixel Size"
         Vertical size of each pixel (in microns).
(*) "Number of Outputs"
         Number of output amplifiers on the chip.
(*) "Frame Transfer"
         Flag indicating if the chip is of frame transfer type.
(*) "Skipper"
         Flag indicating if the chip is of skipper type (multiple read-out
         of the same pixel possible).
```

```
(*) "mpp Mode"
         Flag indicating if mpp read-out mode possible with the chip
         installed.

"Chip Instance"
(*) "Chip Index"
         Read only value: index of the chip under configuration
(*) "Available"
         Chip available
(*) "Flipped"
         Chip flipped
    "Id"
         ESO wide unique identification code of the chip.
    "Date"
         Date of installation (format dd/mm/yy).
(*) "xLocation, yLocation"
         Location (X, Y) of the chip.
         Note: here and in the following we assume that the lowest left
               chip has coordinates (0,0).
    "xGap, yGap"
         Distance (X, Y) between chips in microns (Currently this field
         is ignored)

"Readout"
    "ADC Resolution"
         Number of bits per pixel.
(*) "Wipe Sequence"
         Name of the wipe sequence to be used for periodic wiping
(*) "Wipe Period"
         Number of seconds between consecutive wipes during a periodic
         wipe.

"Output"
(*) "Output Index"
         Read only value: index of the output under configuration
(*) "xLocation, yLocation"
         Pixel location (X, Y) of the output amplifier.
         Note: here and in the following we assume that the lowest left
               pixel of a chip has coordinates (1,1).
(*) "xPrescan, yPrescan"
         Number of prescan columns and rows for each output.
(*) "xOverscan, yOverscan"
         Number of overscan columns and rows for each output.
(*) "Hor/Ver Readout Direction"
         Readout direction.
(*) "Description"
         Description of the output
(*) "Id"
         Identifier of the output

"Image Transfer"
 (*) "Image Extensions"
         When "on" FITS files will be saved using the new image extension
         per port format. When "off" use the traditional FITS file format.

"Modes Configuration"
    Allows selection of the mode to be configured.

"Telemetry Configuration"
    Allows configuration of telemetry.
```

ACTION BUTTONS

```
"Load..."
    Load all values from a user-defined file. Default directory is:
    $INS_ROOT/$INS_USER/COMMON/CONFIGFILES

"Save"
    Write all current values present in the on-line database in the file
    $INS_ROOT/$INS_USER/COMMON/CONFIGFILES/$CCDNAME.dbcfg

"Save as..."
    Write all current values present in the on-line database in a
    user-defined file. Default directory is:
    $INS_ROOT/$INS_USER/COMMON/CONFIGFILES

"Apply"
    Write all current values present in the relative sub-panel in
    the on-line database

"Refresh"
    Read values from the on-line database and display them.
    subpanel.
```

SEE ALSO

fcdConfigMode

```
- - - - - -
Last change:  01/04/04-08:11
```

9.1.5 fcdConfigTel

NAME

fcdConfigTel - panel for configuring CCD telemetry and temperature

MENU BAR

```
"File" menu
    Quit
        exit the application

"Help" menu
    Extended Help. Not implemented yet
```

APPLICATION AREA

```
"Telemetry"
    This check box defines if Telemetry is available for this detector

 To add a new telemetry source, enter each of the fields (including the
 first field which is the location in  the table) and press ADD/MODIFY.

 To change a telemetry source, select the source and modify the values
 dispolayed in the entry areas at the bottom of the panel then press
 ADD/MODIFY.

 To remove a telemetry source, select it from the list and press DELETE.
```

```
Telemetry Fields
---------------

"Index"
   The position of the telemetry source in the table.

"Type"
   Either PULPO (value read from a PULPO unit) or C40 (value
   read from the Detector Head.

"Board"
   Either the PULPO unit number or the board id depending on sensor type.
"Sensor"
   Either the PULPO sensor id or the peripheral id depending type.

"Name"
   Name of sensor (TELE.NAME)
"Id"
   Id of sensor (TELE.ID)

"Units"
   Units the value is represented
"Low", "High"
   Low and high values for monitoring

"Confact"
   Conversion factor from source telemtry value to value stored in OLDB.
   final_value = source_value * ConFact.
```

SEE ALSO

fcdTelemetry

```
- - - - - -
Last change:  01/04/04-08:11
```

9.1.6 fcdConfigMode

NAME

```
fcdConfigMode - panel for FIERA camera mode configuration
```

MENU BAR

```
"File" menu
    Quit
        exit the application

"Help" menu
    Extended Help. Not implemented yet
```

APPLICATION AREA

```
Note: Entries marked with (*) have influence on the camera operations.
      They must be set to the proper value in order to get the camera
      behaving as expected.
      All other parameters have no influence on the camera behaviour;
      they are used only as information to be put in the image FITS
      header.

"Mode Index"
         Read only value: index of the mode under configuration.

"General"
    "Description"
         Description of the mode
    "Gain"
         Value to be put in the FITS file as OUTi.GAIN
    "Speed"
         Speed which is displayed to the user, and is placed in the FITS
         file as READ.SPEED
    "First Output"
         Reference output

"Wipe"
(*) "Sequence"
          Name of the sequence to be used to perform wipe
(*) "Volt table"
          Table of voltages to be applied when wiping (NULL implies no
          change)
    "Wipe Always"
         Flag which indicates that wipe should be performed even when
         a loop of exposures is in progress

"Integration"
(*) "Start Sequence"
          Name of the sequence to be used to perform integration
(*) "During Sequence"
          Name of the sequence to be used during integration
(*) "Volt Table"
          Table of voltages to be applied when Integrating (NULL implies
          no change)

"Readout"
(*) "X/Y Binning Factor  Read Sequence"
          Specifies a hand crafted readout sequence to be used to
          perform a full frame readout for a given X/Y binning combination
(*) "read Sequence"
          Name of the sequence to be used to perform readout
```

```
(*) "Volt Table"
          Table of voltages to be applied when Reading (NULL implies
          no change)

"Useqs"
(*) "Parallel Shift"
          Divide ratio to be applied when performing parallel shift
(*) "Parallel Dump"
          Divide ratio to be applied when performing parallel dump
(*) "Serial Shift"
          Divide ratio to be applied when performing serial shift
(*) "Serial Dump"
          Divide ratio to be applied when performing serial dump
(*) "Serial Read"
          Divide ratio to be applied when performing serial read

"Abs Time"
(*) "Mode"
           What should be synchronised with Absolute time, either
           SPARC scheduled start, or synchronised sequence execution
(*) "Which"
           If a sequence is being synchronised, which sequence Wipe, Int, ..
(*) "Offset"
           Offset by which the SPARC should offset the scheduling. e.g. if
           you want to synchronise an intergration sequence, but the wipe
           is going to take 3s, then you should apply an offset > 3.

"Outputs"
    Buttons indicating which outputs are enabled for this mode

"El/Adu Per Output"
    Stores the DET.OUT.CONAD FITS header parameter for each output

"Readout Nouse Per Output"
    Stores the DET.OUT.RON FITS header parameter for each output
```

ACTION BUTTONS

```
"Apply"
    Write all current values present in the relative sub-panel in
    the on-line database

"Refresh"
    Read values from the on-line database and display them in the relative
    subpanel.

"Close"
    Close the panel (no save).
```

SEE ALSO

```
fcdConfig
```

```
- - - - - -
Last change:  01/04/04-08:11
```

9.1.7 fcdei

NAME

```
fcdei - panel for CCD engineering operations
```

MENU BAR

```
"File" menu
    Quit
        exit the application

"Help" menu
    Extended Help. Not implemented yet
```

APPLICATION AREA

```
"Workstation"
    On the same line are shown:
    - WS Environment name
    - Operational state of CCD sw Ws part

"SLCU"
    On the same line are shown:
    - SLCU Environment name
    - Operational state of CCD sw SLCU part

"CCD System"
    The current operational mode is shown on the same line
    "Change mode ..."
         Pop-up the CCD panel which allows to change the operational mode
         (fcdOpMode).
    "Change Config ..."
         Pop-up the CCD panel which allows to change the camera
         configuration (fcdConfig)
    "Reboot Lcu"
         Reboot remotely the CCD SLCU. It takes a few minutes

"Scan System"
    The current status of the part of scan system used by the CCD system
    is shown (enabled/disabled and active/not active).
    If something does not work, the following buttons should be pushed
    in the same order as presented from top to bottom:
    "Stop CCD Scan"
         Stop the scanning of database values from the CCD SLCU.
    "Clean CCD Scan"
         Clean the list of CCD attributes to be scanned.
    "Conf. CCD Scan"
         Inform the Scan System of the attributes to be scanned for the
         CCD application.
    "Start CCD Scan"
         Start the scanning of database values from the CCD SLCU.

"Utilities"
    "Engineer. Interface"
         Start VLT ccsei utility
```

SEE ALSO

```
ccsei, fcdConfig, fcdOpMode
```

```
- - - - - -
Last change:  01/04/04-08:11
```

9.2 Scripts

In this section the manual pages for the following scripts are presented:

1. CCD stand-alone demo session start-up (*fcdDemo.sh*)
2. CCD stand-alone session start-up (*fcdStart.sh*)

9.2.1 fcdDemo.sh

NAME

fcdDemo.sh - start the FIERA stand-alone sw

SYNOPSIS

fcdDemo.sh [<camera>] [<WS env.>] [<SLCU env.>] [<INS root>]

DESCRIPTION

This shell script pops up the panels used to control a CCD camera in stand-alone. Actions are simulated within the panels. No FIERA program need to run.

<camera> camera name, also root point in DB (default env. var. CCDNAME)
<WS env.> name of workstation environment (default env. var. RTAPENV)
<SLCU env.> name of SLCU environment (default env. var. CCDLENV)
<INS root> INS_ROOT environment variable

ENVIRONMENT

CCDNAME default for camera name (e.g. myccd)
RTAPENV default for WS local environment (e.g. myws)
CCDLENV default for SLCU environment (e.g myfiera)
INS_ROOT default root directory for instrument data

EXAMPLES

fcdDemo.sh myccd myws myfiera $INS_ROOT
Start FIERA sw for camera "myccd", WS environment "myws",
SLCU environment "myfiera". Panels run in simulation.

SEE ALSO

fcdStart.sh

- - - - - -
Last change: 01/04/04-08:11

9.2.2 fcdStart.sh

NAME

```
fcdStart.sh - start the FIERA stand-alone sw
```

SYNOPSIS

```
fcdStart.sh [<camera>] [<WS env.>] [<SLCU env.>] [<INS root>] [<simulation>]
```

DESCRIPTION

```
This shell script pops up the panels from which the operations with
the FIERA stand-alone can be executed.
It also loads the on-line database values for the selected camera.

<camera>     camera name, also root point in DB (default env. var. CCDNAME)
<WS env.>    name of workstation environment (default env. var. RTAPENV)
<SLCU env.>  name of SLCU environment (default env. var. CCDLENV)
<INS root>   INS_ROOT environment variable
<simulation> 1 = panels are entered in simulation mode: no action done.
```

ENVIRONMENT

```
CCDNAME  default for camera name (e.g. myccd)
RTAPENV  default for WS local environment (e.g. myws)
CCDLENV  default for SLCU environment (e.g myfiera)
INS_ROOT default root directory for instrument data
INS_USER default SYSTEM
```

EXAMPLES

```
fcdStart.sh myccd myws myfiera $INS_ROOT 1
Start FIERA sw for camera "myccd", WS environment "myws",
SLCU environment "myfiera". Panels run in simulation.
```

SEE ALSO

```
fcdDcsStart.sh fcdDcsStop.sh
```

```
- - - - - -
Last change:  01/04/04-08:11
```

__oOo__