

Local Run Control

version gui-v2.5-pro

The local run control is intended to be used when running in standalone mode, i.e. with only the TPC readout. This is the case e.g. if one wants to do a pedestal run, local data taking, or some test. The local run control user interface is shown in figure LCR-1. In figure LCR-2 is the program flow and communication with the ilcserver shown. It is written in Java, and use a thread for receiving information from the ilcserver. The IP-address and port on the ilcserver machine is read from the file **ilcdaq.txt**, to be placed in the running directory, with DESY settings:

SERVER 131.169.184.203

SERVER.PORT 2002

PORT 8001

- SERVER xxxx.xxxx.xxxx.xxxx – IP-number of the ilcserver
- SERVER.PORT n – port number that the ilcserver listens on.
- PORT n – port number that the local run control listens on.

While the DAQ is stopped can one switch on/off (*PowOn*/*PowOff*) the front end cards. It is not possible to change the settings, this has to be done in the configuration file of the RCU (in rcu-<id>.cfg item RCU.POWER and RCU.READOUT). The colors indicate the knowledge of the power status:

- light blue: FEC should be off according to the configuration file. Hardware status unknown.
- dark blue: FEC should be on according to the configuration file. Hardware status unknown.
- red: FEC is off according to the hardware status flag in the RCU.
- green: FEC is on according to the hardware status flag in the RCU.
- others: status of FEC according to RCU is not the same as found on the ALTRO bus.

While the DAQ is stopped can one load the settings for the PCA16 chips, *PcaLoad*'. In contrast with the power can one change these settings from the run control. These settings are when loaded also stored in a configuration file by the readout program. It is not possible to read back the values loaded into the PCA16, what is shown is always the last known content of the configuration file.

Since it takes some time to power on and load the PCA settings is this not done automatically, it must be done manually in the way described, before any DAQ handling is started. The reason is to avoid delays in the run handling for local/global run control for things not often changed.

There are lookup tables loaded for each ALTRO channel. These tables takes some time to load. One can choose to load them at PowerOn, check the box “Load lookup tables at power on” (which is checked by default) or at StartDAQ, check the box “Load lookup tables at Start DAQ” (not checked by default). To avoid loading the tables at each StartDAQ, do StopDAQ and uncheck the box “Load lookup tables at Start DAQ”, and do StartDAQ again.

There are three run types, to be specified before start of run:

- physics – normal data taking, in this mode must the readout program configure a physics trigger
- pedestals – in this mode must the readout program configure a pulser trigger/random trigger.
- test – not implemented

Pedestal subtraction and zero suppression can be specified, zero suppression will force pedestal subtraction. For pedestal runs uncheck these.

To enable the readout to send events for the monitoring task must the value in '*Monitor*' be nonzero. After a request for an event from the monitoring program is this the number of events taken before a new request is served, i.e. a sort of downscaling. There are two options 'Read events' and 'Monitor events'. If 'Read events' then the run is automatically stopped when this number of events has been read. If 'Monitor events' then the run is automatically stopped when this number of events has been sent to the monitoring task.

By checking '*Logging*' will a data file be written by the readout program.

A run comment can be added to the run log file, which is written by the readout program. It will be sent when pressing '*--- Write run comment ---*', it is not written at start of run but immediately. This is not (yet) put into the data stream, so it has no real use at the moment.

In the message window is the communication with the ilcserver shown.

The screenshot displays the Local Run Control user interface, which is organized into several functional sections:

- Power settings:** Located at the top, it features three rows of status indicators for RCU0, RCU1, and RCU2, each with 32 colored boxes (red, green, blue, cyan). To the right are 'PowOn' and 'PowOff' buttons. Below the indicators are checkboxes for 'Load lookup tables at power on' (checked) and 'Load lookup tables at Start DAQ' (unchecked).
- PCA settings:** Below power settings, it includes checkboxes for 'Polarity' (checked), 'Shutdown' (unchecked), and 'Preamp enable' (unchecked). It also has input fields for 'Gain' (0), 'Shaper' (1,000), and 'Decay time', followed by a 'PcaLoad' button.
- Control Buttons:** A vertical column on the left contains buttons for 'DAQ Active' (cyan), 'Stop DAQ' (green), 'Start run' (green), 'Pause' (cyan), 'Run stopped' (red), 'Status' (green), and 'SCRIPT' (cyan).
- Log Window:** A large text area in the center displays a log of system messages, including timestamps and status reports for the DAQ and power systems.
- Run Comment:** A text input field labeled 'Run comment (max 240 characters):' is positioned below the log window.
- Run Controls:** At the bottom, there is a section for 'Run type' (Physics selected), 'Run mode' (Pedestal subtraction and Zero suppression checked), and 'Logging' (checked). It also includes a 'Monitor' section with a dropdown menu set to 'Read events' and a numerical input field set to '1'.

Figure LCR-1: Local Run Control user interface

The readout can be in several states: the DAQ is running/stopped, a run is running/stopped/paused. These states is controlled from the buttons on the local run control display. The meaning of the buttons change depending on the state. Pause run is in the current version disabled.

- Start DAQ – this will tell the readout program to open DRORC devices, setting up memories ...
- Stop DAQ – close devises and disable memories.
- Start run – start a run
- Pause run – pause an active run (DISABLED in version > 2.4)
- Continue run – continue an active run
- Stop run – stop an active run
- SCRIPT – load a script to be executed

A green button can be pressed, a red is disabled, and a blue indicates the system is busy. The status of the readout can be requested with the 'Status' button. At a regular interval is the readout task sending the number of events taken. In case an error is encountered the 'Status' button is turned into red.

SCRIPT is a feature to load a script. On load is the syntax checked and if no errors found then the Start DAQ button is turned into a Start SCRIPT, and SCRIPT to abort script. The script file must have the extension .run. The script is executed line by line. The syntax is:

```

SETTINGS                                Initial setup
ERROR.RETRY 30                          Number of times to retry in case of error (NOT WORKING FOR ALL ERRORS)
LOOKUP.POW  1                           load lookup tables at power on FEC
LOOKUP.DAQ  0                           load lookup tables at Start DAQ
FEC.ON                                           Power on active FEC
END
# *****
# RUN 1 GAIN 0 SHAPER 0 DAC 0
#
SETTINGS                                Setup for run
PCA.POLARITY  1                          PCA polarity bit
PCA.SHUTDOWN  0                          PCA shtdown bit
PCA.PREAMP    0                          PCA preamp enable bit
PCA.GAIN      0                          PCA gain bits (0-3)
PCA.SHAPER    0                          PCA shaper bits (0,1,3,7)
PCA.DAC       0                          PCA decay time DAC setting (0-2500)
PCA.LOAD                                             Load settings to PCAs
END
RUNNING                                Run parameters
#RUN.PHYSICS                                Do a physics run
RUN.PEDESTALS                              Do a pedestal run
DAQ.START                                  Start DAQ

```

#RUN.EVENTS	500	Number of events to read
RUN.MEVENTS	500	Number of events to monitor
RUN.MONITOR	1	Send Nth (here “every”) event for monitoring
RUN.LOGGING	1	Log the data
RUN.PEDSUB	0	Pedestal subtraction
RUN.ZEROSUP	0	Zerosuppression
RUN.START		Start Run
DAQ.STOP		Stop DAQ
END		

The protocol between the local run control and the ilcserver are text strings sent over the network, as shown in figure LCR-2. When the run control starts up it sends *SRV to the ilcserver, and starts a thread listening for incoming packets from the ilcserver. The ilcserver sends back *STATUS when receiving the *SRV command. At Start DAQ is *START sent, after receiving *STATUS, is *POW sent to retrieve the current power settings of FEC, and finally *PCA to get the PCA16 settings used. *POW and *PCA is also sent when pressing the 'PowOn'/'PowOff' and 'PcaLoad'. The readout sends back the same string with status arguments. 'Stop DAQ' will send *STOP, 'Start Run' *SOR, 'End Run' *EOR, 'Pause' *PAUSE, 'Continue' *CONT, for these will the readout send back *STATUS.

***CONT** – *continue a paused run.*

no arguments, ilcserver returns *STATUS.

***EOR** – *end an active run.*

no arguments, ilcserver returns *STATUS.

***PAUSE** – *pause an active run.*

no arguments, ilcserver returns *STATUS.

***PCA** – *load/get PCA16 settings.*

if no argument get the settings used. Arguments are

*PCA SR n DAC n

- SR n – n is the pattern for the shift register setting individual bits to PCA
- DAC n – n is the DAC value to set for the decay time

***POW** – *power on/off front end cards.*

It has one argument when sent from run control:

*POW n – n=0, get current status, n=1 power on active FEC, n=2 power off all FEC, n=5 power on and load lookup tables.

The readout sends back:

*POW S0 n0 RCU0 m0 S1 n1 RCU1 m1 S2 n2 RCU2 m2 S3 n3 RCU3 m3 ERR n

Sn nX = Source of information RCU id n, nX=0 from hardware, nX=1 from file, nX=2 no information

RCUn mX = bit pattern giving status for RCU id n, bit on = power is on, bit off = power is off.

***SOR**- *start of run.*

no arguments, ilcserver returns *STATUS.

***SRV** – *sent by local run control to start communication.*

no arguments, ilcserver returns *STATUS.

***START** – *start data acquisition.*

***START CONTROL n MODE n TYPE n LOAD n**

- CONTROL n – n=1 I'm the local run control (needs to be rethought of!!).
- MODE n – n=1 physics, n=2 pedestals, n=3 test (not yet used).
- TYPE n – n=0 no pedestal subtraction/zero suppression, n=1 pedestal subtraction, N=3 pedestal subtraction and zero suppression.
- LOAD bit 0: load lookup tables at power on, bit 1: load lookup up tables at StartDAQ

***STATUS** – *current status.*

*STATUS is sent without any arguments from the run control to get the current status, it has a number of arguments as received from the ilcserver.

***STATUS DAQ n RUN n LOG n MON n EVT n MODE n RUNNB n ERR n**

- DAQ n – DAQ status; n=0 DAQ is stopped, n=1 DAQ is active
- RUN n – RUN status; n=0 RUN is stopped, n=1 RUN is running, n=2 RUN is paused
- LOG n – Logging status; n=0 no data file written, n=1 data file written
- EVT n – Events taken
- TYPE n – run type; n=0 no pedestal subtraction/zero suppression, n=1 pedestal subtraction, n=3 pedestal subtraction and zero suppression.
- MODE n – run mode; n=1 physics, n=2 pedestals, n=3 test
- RUNNB n – n = the run number as given by the readout program
- ERR n – error status; n=0 no error, n>0 some obscure error number
- LOAD n – returned only when lookup tables are loaded

***STOP** – *stop data acquisition.*

no arguments, ilcserver returns *STATUS.

***UPD** – *update information from readout.*

sent by the readout at a regular interval.

***UPD EVT n**

- EVT n – n=number of events taken

FILES

DaqPanels.java	JAVA source code
ilcdaq.txt	configuration file

readme.txt

updates made

genscript.sh

script to generate a .run script

COMPILE AND RUN

```
javac DaqPanels.java
```

```
java DaqPanels
```

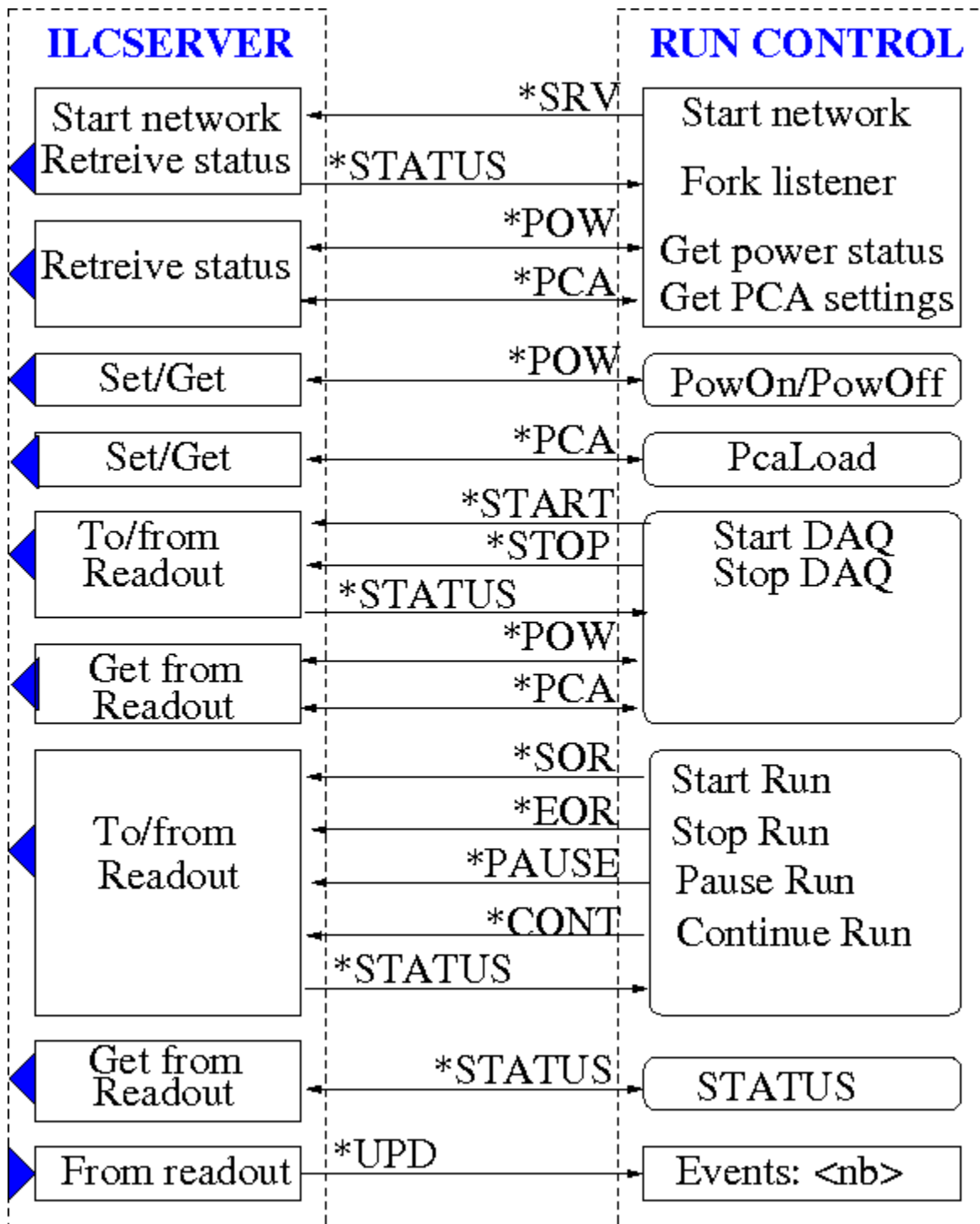


Figure LCR-2: Local Run Control communication with ilcserver.

Changes

2.1 Recognize the reporting *UPD from readout program

2.2 Remove waiting for OK after some commands

replace client port 2002 with configuration from ilcdaq.txt

SERVER.PORT n

2.3 update with other colors on power status

possibility to run scripts

2.4 remove the possibility to pause run

20090206 *POW: itemvalue = (int) Long.parseLong(st.nextToken(),16);

2.5

checkboxes: load lookup tables at power on or startdaq

*POW 5

*START ... LOAD n

*STATUS LOAD n

script: RUN.LOGGING, RUN.PEDSUB, RUN.ZEROSUP, LOOKUP.POW, LOOKUP.DAQ