

Local Run Control

version gui-v2.8-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.
- CLIENT.PORT n – port number that the local run control listens on.
- DB.SERVER xxxx.xxxx.xxxx.xxxx – IP number of data base server
- DB.PORT n – port number that the data base server listens to
- TEMP.PORT n – port number that the program use to listen to temperature server

While the DAQ is stopped can one switch on/off (*PowOn*/*PowOff*) the front end cards. It is possible to change the settings in the topmenu FEcs, this can also 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.
- yellow/grey: the RCU did not respond
- 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*', one can 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 – memory test runs

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.

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

The temperature on the FECs are monitored by a temperature sensing system (described elsewhere). The GUI communicates with the temperature server thorough the network. The last read highest temperature and the temperature sensor identifier is displayed. There are two alarm levels, a low warning level (text switched to blue color) and alarm (red). If the temperature exceeds the alarm limit is there a command sent to DAQ to power off. Alarm messages are written in the '*Temperature alarms:*' message window.

DRORCs
RCUs
FECs
ALTRO
Tests

Power settings

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
RCU0
RCU1
RCU2
RCU3

PowOn
PowOff

☒ Load lookup tables at power on
☐ Load lookup tables at Start DAQ
Alarm

PCA settings

☒ Polarity
☐ Shutdown
☐ Preamp enable
27 mV/fC
Gain
120 ns
Shaper
0
Decay time
Pcaload

DAQ Active
Stop DAQ
Running
Pause
Stop run
Status
SCRIPT

[2010-03-04 14:26:14] SRV:*PCA SR 134 DAC 0 ERR 0
[2010-03-04 14:26:15] Stopping DAQ
[2010-03-04 14:26:15] SRV:*STATUS DAQ 0 RUN 0 LOG 0 MON 1 EVT 431 TYPE 3 MODE 1 RUNNB 16
[2010-03-04 14:26:18] Starting DAQ
[2010-03-04 14:26:18] *START CONTROL 1 MODE 1 TYPE 3 LOAD 2
[2010-03-04 14:26:18] SRV:*STATUS DAQ 1 RUN 0 LOG 0 MON 1 EVT 431 TYPE 3 MODE 1 RUNNB 16
[2010-03-04 14:26:18] Retrieving power status...
[2010-03-04 14:26:18] SRV:*POW S0 1 RCU0 C0000 P0 0 S1 1 RCU1 440000 P1 0 S2 1 RCU2 48 P2 0
[2010-03-04 14:26:18] Retrieving PCA settings...
[2010-03-04 14:26:18] SRV:*PCA SR 134 DAC 0 ERR 0
[2010-03-04 14:26:19] Starting run
[2010-03-04 14:26:19] SRV:*STATUS DAQ 1 RUN 1 LOG 0 MON 1 EVT 0 TYPE 3 MODE 1 RUNNB 1675
[2010-03-04 14:26:21] Retrieving status...
[2010-03-04 14:26:21] SRV:*STATUS DAQ 1 RUN 1 LOG 0 MON 1 EVT 19 TYPE 3 MODE 1 RUNNB 167
[2010-03-04 14:26:22] Retrieving status...
[2010-03-04 14:26:22] SRV:*STATUS DAQ 1 RUN 1 LOG 0 MON 1 EVT 27 TYPE 3 MODE 1 RUNNB 167
[2010-03-04 14:26:32] SRV:*UPD EVT 125

Temperature alarms:
[2010-03-04 14:25:45] ALARM level: 1 sensor: 1 value: 44

[2010-03-04 14:26:36] Sensor with highest value: 1 value= 44

Events: 125

Run:16752
[2010-03-04 14:26:19]

Exit

Run type:
Run mode:

☒ Physics
☐ Pedestals
☐ Test

☒ Pedestal subtraction
☒ Zero suppression
☐ Logging

Monitor
1
Read events
0

Figure LCR-1: Local Run Control user interface

Topmenus

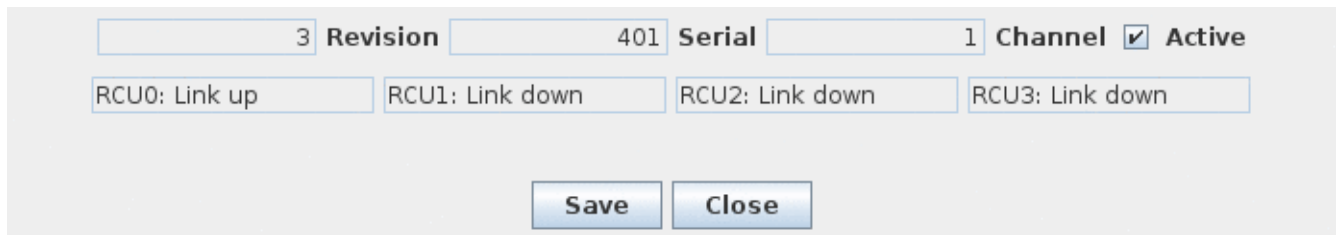
DRORCs

DRORCs
RCUs
FECs
ALTRO
Trigger
Tests

Find DRORCs
Active DRORC RCU 0
Active DRORC RCU 1
Active DRORC RCU 2
Active DRORC RCU 3
☒ Load lookup tables at power on
☐ Load look

Find DRORCs – this will rescan the hardware for switched on RCUs.

Active DRORC will allow you to see links, their serial number, if there is any connected and switched on RCU, and if the DRORC channel is to be active in the readout.



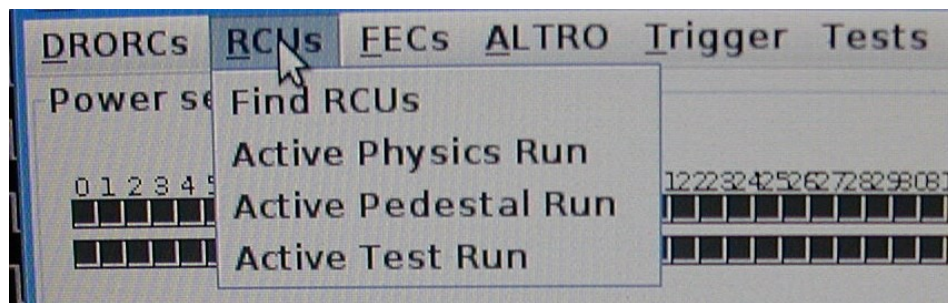
3 Revision 401 Serial 1 Channel ☒ Active

RCU0: Link up RCU1: Link down RCU2: Link down RCU3: Link down

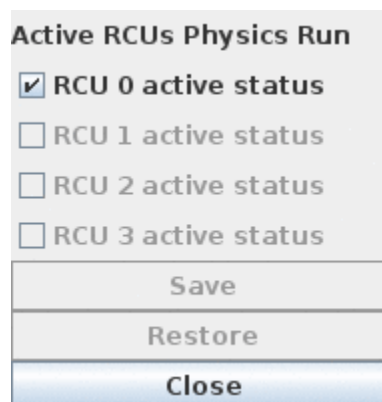
Save Close

RCUs

Find RCUs will check if the active RCUs are responding. If the power settings is shown in yellow/gray means that there is no response from the RCU. Can be used after a power on of the low voltage to check if the RCU has finished booting.



Active - select switched on RCUs to be readout for physics run, pedestal run, test run.



Active RCUs Physics Run

☒ RCU 0 active status

☐ RCU 1 active status

☐ RCU 2 active status

☐ RCU 3 active status

Save

Restore

Close

FECs

One submenu for each RCU. Selecting an RCU will show a window where one can select the FECs to power on.

☐ FEC 0	☐ FEC 16
☐ FEC 1	☐ FEC 17
☐ FEC 2	☐ FEC 18
☐ FEC 3	☒ FEC 19
☐ FEC 4	☐ FEC 20
☐ FEC 5	☐ FEC 21
☐ FEC 6	☐ FEC 22
☐ FEC 7	☐ FEC 23
☐ FEC 8	☐ FEC 24
☐ FEC 9	☐ FEC 25
☐ FEC 10	☐ FEC 26
☐ FEC 11	☐ FEC 27
☐ FEC 12	☐ FEC 28
☐ FEC 13	☐ FEC 29
☐ FEC 14	☐ FEC 30
☐ FEC 15	☐ FEC 31
Save	Restore
Close	

ALTROs

Submenu settings -> physics, pedestal, test

Here one can change the zero suppression threshold and number of time samples to take.

8	Threshold	1000	Samples
Save		Close	

Tests

Memory -> setup for a ALTRO memory test (expert only).

☒ Enable
 ☐ None
 ☐ Step

☒ Reverse 8-15
 ☐ Reverse 0-7
 ☐ 155

☐ 2AA
 ☐ Peaks
 ☐ Table

☐ 0
 ☐ 3FF
 ☐ Walking one

☐ Random

Readout control

The readout can be in several states: the DAQ is running/stopped, a run is running/stopped/paused. These states are controlled from the buttons on the local run control display. The meaning of the buttons changes 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 devices 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 the readout task sends 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 the syntax is checked and if no errors are 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 shutdown 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
LOOKUP.TABLE	0	Use a specific lookup table
RUN.PEDESTALS		Do a pedestal run
DAQ.START		Start DAQ
RCU.ACTIVE	0 1 1 0 2 0 3 0	Active RCUs <rcuid> <flag>
#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.

***RORCFIND** – rescan hardware for switched on RCUs – no reply expected

***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

Temperature alarms

If the Lund Temperature Monitoring hardware is installed on the FECs then the temperature is shown in the windows 'Temperature alarms'. The upper window show alarm history, the lower the currently highest temperature. The color code is:

green – temperature below low alarm level

blue – temperature warning, above low alarm level and below high alarm

red – above high alarm level. The system tries to switch off the power to the FECs in an orderly way: stop any active run, stop DAQ, and send power off command to the DAQ.

The program listens on the port given by TEMP.PORT in the configuration file for the command:

***TEMP A <pattern> S <sensor> T <temperature>**

A <pattern> = Alarm pattern, bit 0 – hard low, 1 – hard high, 2 – soft low, 3 – soft high

S <sensor> = Sensor with highest temperature

T <temperature> = Highest temperature in degrees (integer)

hard alarm is not implemented. Soft alarm means software is responsible for all alarm actions, e.g. power off.

The temperature system is described elsewhere.

DATA BASE

When asking for and sending parameters is the run control communicating with a data base handler. Note that the data base server updates the readout configuration files. The changes will be read in at next start DAQ or start Run depending on the parameter.

***ALTRO TYPE <runtype>** retrieve information

***ALTRO TYPE <runtype> TH <threshold> SA <samples>** settings

***DRORC ID <rcuid>** retrieve information

***DRORC ID <rcuid> RE <revision> SE <serial> ST <status> CH <channel> LU <linkstatus>**

*DRORC ID <rcuid> ST <status>	information returned
*RCU ID <rcuid>	update status not active/active
*RCU ID <rcuid> R <fec bitpattern>	retrieve information
*RCU ID <rcuid> R <fec bitpattern>	information returned
*RCUACTIVE TYPE <runtype>	update information
*RCUACTIVE TYPE <runtype> FL <activeflag> ST <status>	retrieve information
*RCUACTIVE TYPE <runtype> FL <activeflag>	settings status= of DRORC channel
*MEMTEST	update information
*MEMTEST LU <lookup table> EN <enable>	retrieve information
*MEMTEST LU <lookup table> EN <enable>	information returned
	update information

FILES

DaqPanels.java	main JAVA source code
altroDialog.java	ALTRO setup dialog
dbHandler.java	Data base communication
drorcDialog.java	DRORC setup dialog
fecDialog.java	FEC setup dialog
memoryDialog.java	Memory test setup dialog
rcuDialog.java	RCU setup dialog
ScriptFilter.java	script dialog
tempmonHandler.java	temperature server communication
ilcdaq.txt	configuration file
readme.txt	updates made
genscript.sh	script to generate a .run script

COMPILE AND RUN

```
javac DaqPanels.java
java DaqPanels
```

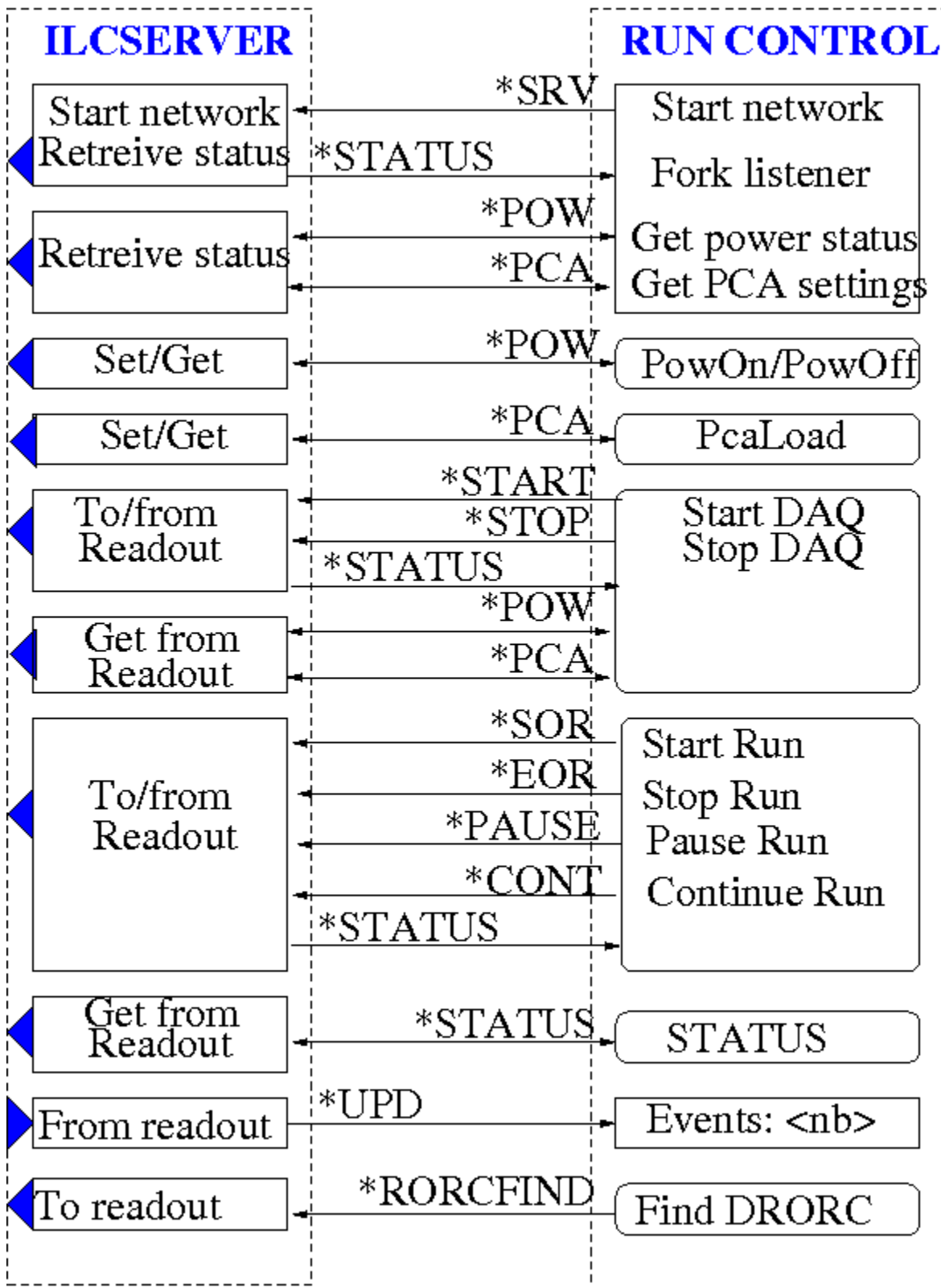


Figure LCR-2: Local Run Control communication with ilcserver.

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

2.7

data base handling for DRORC,RCU,ALTRO etc.. in topmenu

temperature monitoring

gain/shaping with values instead of 0,1,2...

script: run with a part of RCUs, memory test

2.8

Find RCUs