LERF LLRF Engineer's Guide

Last updated 5/22/2019, 10:30 PDT S. Hoobler

Contents

1. Help

Wesley Moore maintains the LERF control system infrastructure and can help with your accounts, setup, etc. (wmoore@jlab.org)

Hugo Slepicka (slepicka@slac.stanford.edu) is a good resource for questions about the LinuxRT OS and its python installation.

2. CPUs, IOCs, and Servers

One LinuxRT CPU is connected to each pair of cryomodule racks. One IOC runs on that CPU and interfaces to all the chassis in those two racks.

Any task that requires access to the LLRF private network must be performed on the appropriate LinuxRT CPU. Those CPUs have a limited set of python modules installed. Tasks that require other python modules must be performed on the user Linux servers/workstations; from there you will not have access to the private network and thus must use EPICS Channel Access.

LCLS-II Cryomodule Name	_LinuxRT CPU Node Name*_	_EPICS IOC Name*_	*JLab Cryomodule Number*
ACCL:L1B:0200	lcls-llrfcpu01	sioc-l1b-rf01	1
ACCL:L1B:0300	lcls-llrfcpu02	sioc-l1b-rf02	2

*CPU Node Name is referred to as <cpuname>in the commands shown below.

EPICS IOC Name is referred to as <iocname> in the commands shown below.

Node	*Type*	*Tasks*
lclsapp1	Linux server	For software installation and testing
lcls01, lcls02, lcls03	Linux workstation	

Please contact Wesley (above) for instructions on which servers/accounts to use for your work—it may be different for software installation vs testing.

3. Chassis IPs

These are the IP addresses used in the LLRF internal network. They are the same for each cryomodule.

Rack	*Chassis*	*IP*
Cavities 1-4 (aka Rack A)	RES	192.168.0.100
Cavities 1-4 (aka Rack A)	RFS1 (cavities 1,2)	192.168.0.101
Cavities 1-4 (aka Rack A)	RFS2 (cavities 3,4)	192.168.0.102
Cavities 1-4 (aka Rack A)	PRC	192.168.0.103
Cavities 5-8 (aka Rack B)	RES	192.168.0.200
Cavities 5-8 (aka Rack B)	RFS1 (cavities 5,6)	192.168.0.201
Cavities 5-8 (aka Rack B)	RFS2 (cavities 7,8)	192.168.0.202
Cavities 5-8 (aka Rack B)	PRC	192.168.0.203

4. Rack and Chassis EPICS PV Prefixes

These are the PV prefixes for the racks and chassis. These are referred to as <prefix> in the commands shown later in this document.

ACCL:L1B:0200 entity	*PV Prefix*	*ACCL:L1B:0300 entity*	*PV Prefix*
Rack A	ACCL:L1B:0200:RACKA	Rack A	ACCL:L1B:0300:RACKA
Rack A RFS1	ACCL:L1B:0200:RFS1A	Rack A RFS1	ACCL:L1B:0300:RFS1A
Rack A RFS2	ACCL:L1B:0200:RFS2A	Rack A RFS2	ACCL:L1B:0300:RFS2A
Rack A PRC	ACCL:L1B:0200:PRCA	Rack A PRC	ACCL:L1B:0300:PRCA
Rack A RES	ACCL:L1B:0200:RESA	Rack A RES	ACCL:L1B:0300:RESA
Rack B	ACCL:L1B:0200:RACKB	Rack B	ACCL:L1B:0300:RACKB
Rack B RFS1	ACCL:L1B:0200:RFS1B	Rack B RFS1	ACCL:L1B:0300:RFS1B
Rack B RFS2	ACCL:L1B:0200:RFS2B	Rack B RFS2	ACCL:L1B:0300:RFS2B
Rack B PRC	ACCL:L1B:0200:PRCB	Rack B PRC	ACCL:L1B:0300:PRCB
Rack B RES	ACCL:L1B:0200:RESB	Rack B RES	ACCL:L1B:0300:RESB

5. Configuration Control

Any changes to critical software/firmware must go through an approval and tracking process. For LLRF, this includes:

- RFS/PRC/RES firmware*
- lcls2_llrf software^
- EPICS ‘RF’ application (app booted by IOCs)^
  - Note that any change to FEED EPICS support also requires a new release of RF
- Some specific scripts/configuration files (section 6.c,d)~

If you need to make changes:

1. Send an email describing your planned changes to this group:
   Curt Hovater (hovater@jlab.org), Gary Croke (gcroke@jlab.org), Ramakrishna Bachimanchi (bachiman@jlab.org)
2. Curt or Rama will create a JLab ATLis ticket.
3. Gary will review and approve.
4. Curt or Rama will work with Main Control to give you access.
5. Once you are done, notify the group and send the new version information:
   * for firmware, we send file name and git commit ID (as reported by FEED)
   ^ for lcls2_llrf and RF EPICS app, we send the git tag name (gitlab and SLAC repos, respectively)
   ~ for these individual files, we send the SLAC repo CVS version number

6. File System Locations and Info

Probably need to know

a. lcls2_llrf:

/usr/local/lcls/package/lcls2_llrf (with submodules)

This is always a tagged 'production' version and is not treated as a sandbox. You may need to temporarily modify something here so that an EPICS wrapper script will execute your new version. Once testing is done, please commit/tag your changes. For more general testing/development, please check out the repo into your own working area. See Section 11 if you need to make changes here.

b. Kintex bitfiles:

/usr/local/lcls/tools/FEED/firmware/prc
/usr/local/lcls/tools/FEED/firmware/res_ctl
In each of these subdirectories, the 'current' symbolic link points to the version that is loaded by the rack-checkout and resonance-init scripts. See Section 12 if you need to use a new bitfile.

Hopefully won't need to know

c. FEED launcher configuration file (needed for rack checkout, pulse control, cavity ramp, etc.):
/usr/local/lcls/tools/FEED/config/rf_control_launcher_LERF.conf

d. Various wrapper scripts used for rack checkout/initialization:
/usr/local/lcls/tools/scripts/
    rfInitResLcls2.sh
    rfInitResLerf.sh
    rfRackTestCommon.sh
    rfRackTestLerf.sh

e. EDM files:
/usr/local/lcls/tools/edm/display
'lerf' or 'llrf' subdirectory

f. FEED EPICS module:
In addition to package/lcls2_llrf, FEED is also installed in the EPICS module area. In this location, it is only used to provide EPICS libraries and databases to the RF EPICS IOC application.
/usr/local/lcls/epics/R3.15.5-1.0/modules/FEED/<FEED-tag-name>
See Section 13 if you need to expose new FW registers via EPICS or to make other changes to FEED EPICS support.

g. RF EPICS IOC application:
   This is the application booted by the 2 LLRF IOCs:
   /usr/local/lcls/epics/iocTop/RF/<RF-tag-name>
   The 'current' symbolic link in the RF directory points to the in-use version. See Section 13 if you need to release a new version of RF.

7. Executing Non-Channel-Access Scripts

1. Halt communication between EPICS and the FPGA. This can be done per-chassis or per-rack
   ```
   caput <prefix>:CTRL_HALT 1
   ```

2.
   ```
   ssh laci@<cpuname>
   ```
   Hit carriage return

3. cd to your desired directory and do your work
   If 'python' is not recognized (which may happen when entering a new shell):
   ```
   source /usr/local/lcls/epics/iocCommon/facility/GoPythonLinuxRTEnvs.sh
   ```

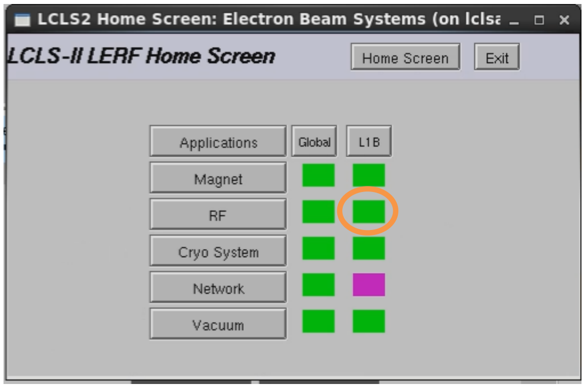
   Most areas of the file system are read-only from these CPUs.
   You can write files to: /data/<cpuname>

4. When done, resume EPICS<->FPGA communication:
   ```
   caput <prefix>:CTRL_RESET 1
   ```

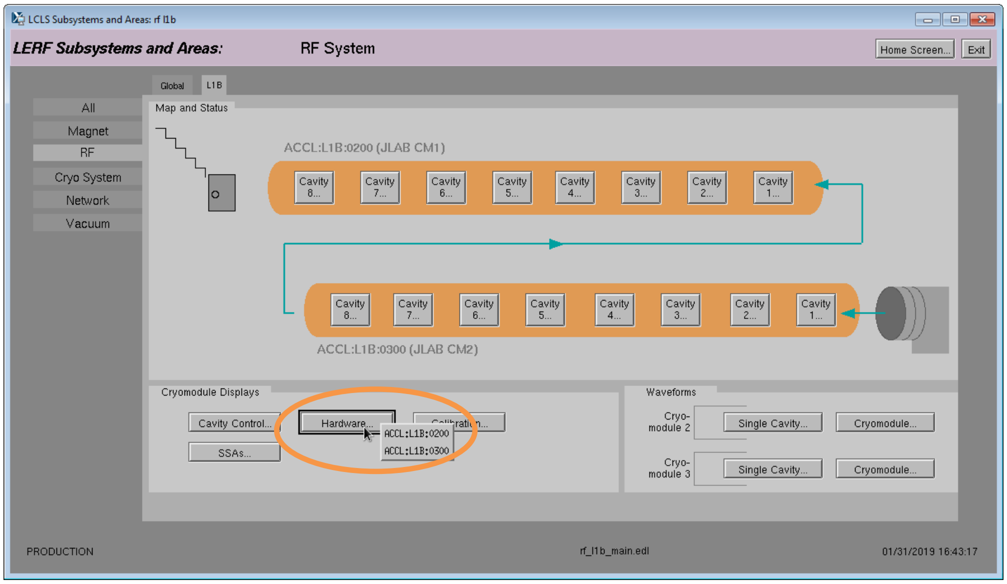
8. EPICS LLRF Hardware Diagnostics

To get to the EPICS LLRF diagnostic screens:

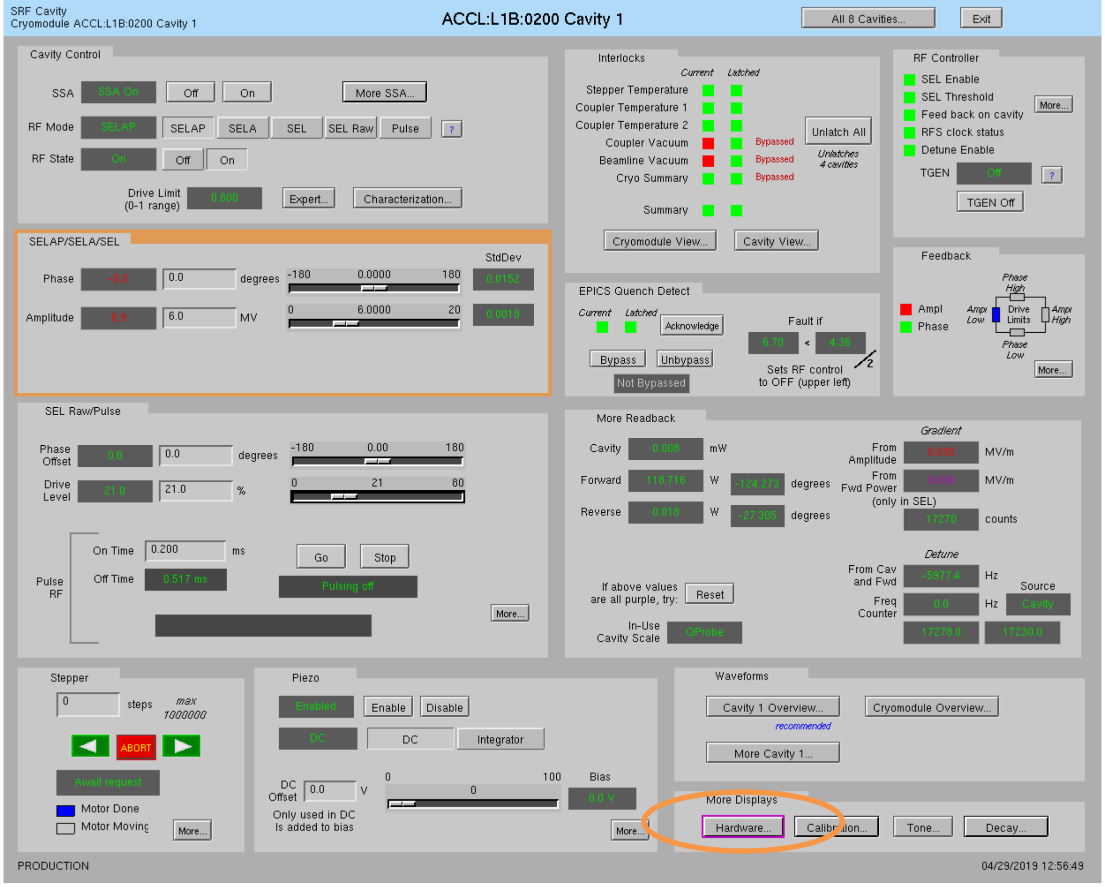
Type lerfhome&



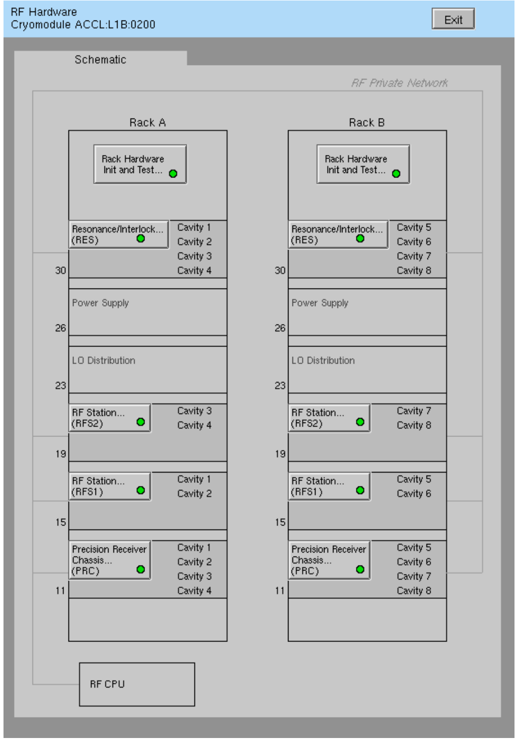
Click on box intersecting RF and L1B



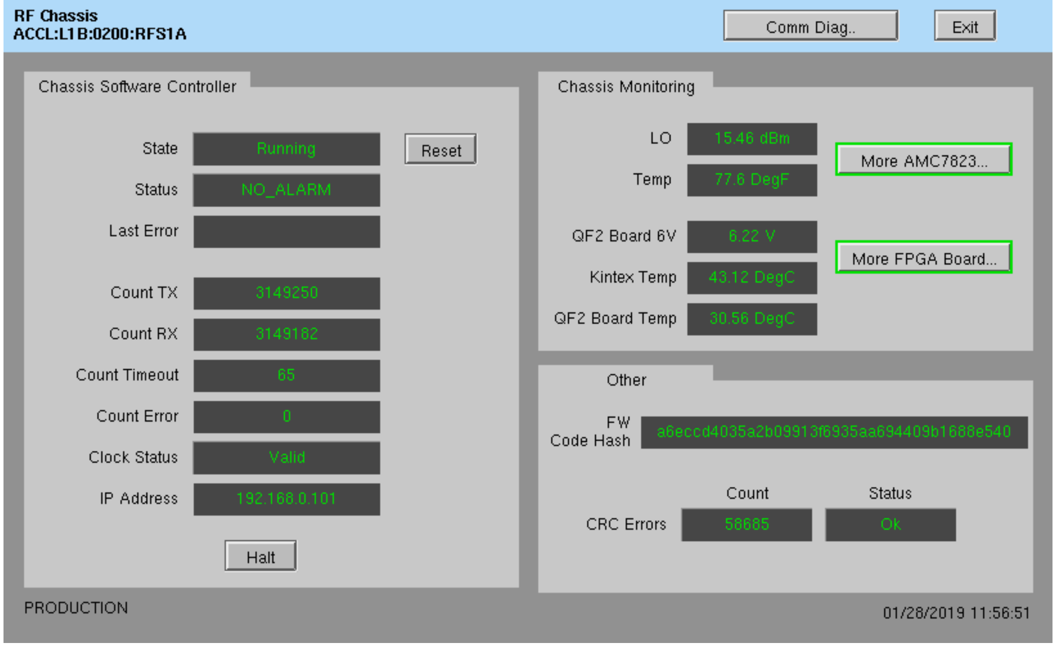
Click Hardware… and select cryomodule of interest from drop-down menu



Or from individual cavity display, click Hardware…


RF Hardware
Cryomodule ACCL:L1B:0200

Note that at LERF, RFS2 is above RFS1 in the rack. At SLAC, they are reversed.


RF Chassis
ACCL:L1B:0200:RFS1A

Chassis Software Controller

State: Running
Status: NO_ALARM
Last Error:
Count TX: 3149250
Count RX: 3149182
Count Timeout: 65
Count Error: 0
Clock Status: Valid
IP Address: 192.168.0.101

Chassis Monitoring

LO: 15.46 dBm
Temp: 77.6 DegF
QF2 Board 6V: 6.22 V
Kintex Temp: 43.12 DegC
QF2 Board Temp: 30.56 DegC

Other

FW Code Hash: a6eccd4035a2b09913f6935aa694409b1688e540

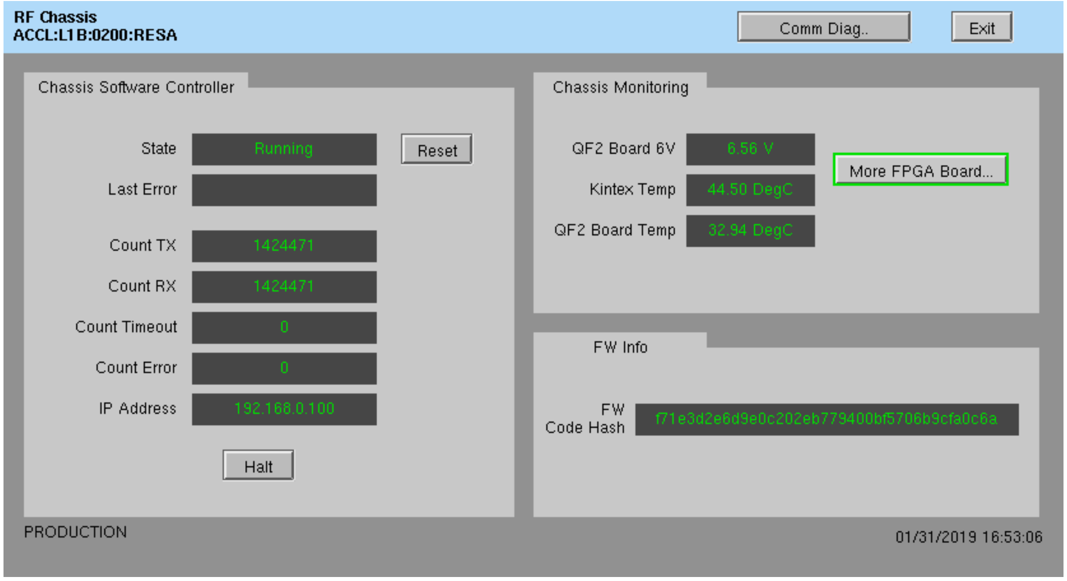
	Count	Status
CRC Errors	58685	Ok

PRODUCTION

01/28/2019 11:56:51

RFS/PRC display:

AMC7823 chip monitoring
QF2 board monitoring
CRC errors



RF Chassis
ACCL:L1B:0200:RESA

Chassis Software Controller

State: Running

Last Error:

Count TX: 1424471

Count RX: 1424471

Count Timeout: 0

Count Error: 0

IP Address: 192.168.0.100

Chassis Monitoring

QF2 Board 6V: 6.56 V

Kintex Temp: 44.50 DegC

QF2 Board Temp: 32.94 DegC

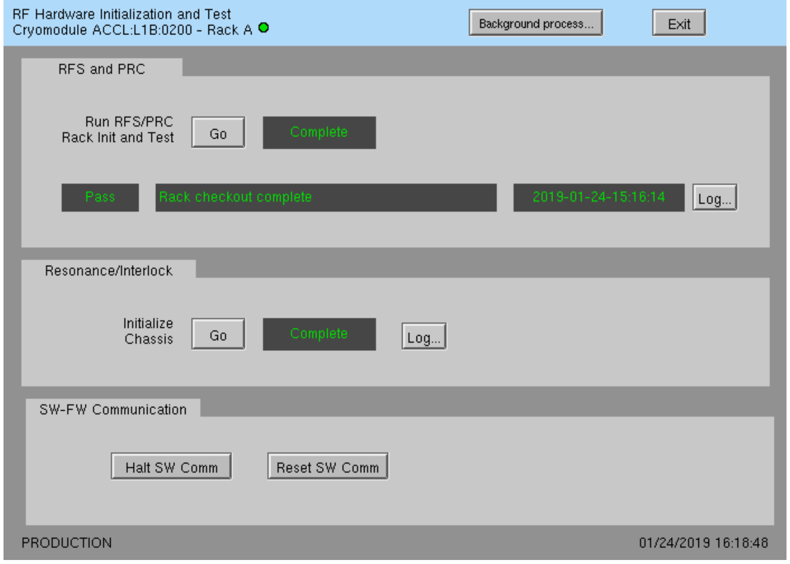
FW Info

FW Code Hash: f71e3d2e6d9e0c202eb779400bf5706b9cfa0c6a

PRODUCTION

01/31/2019 16:53:06

Res/Intlk display does not have AMC7823 monitoring nor CRC errors



RF Hardware Initialization and Test
Cryomodule ACCL:L1B:0200 - Rack A ●

RFS and PRC

Run RFS/PRC Rack Init and Test: Complete

Pass | Rack checkout complete | 2019-01-24-15:16:14

Resonance/Interlock

Initialize Chassis: Complete

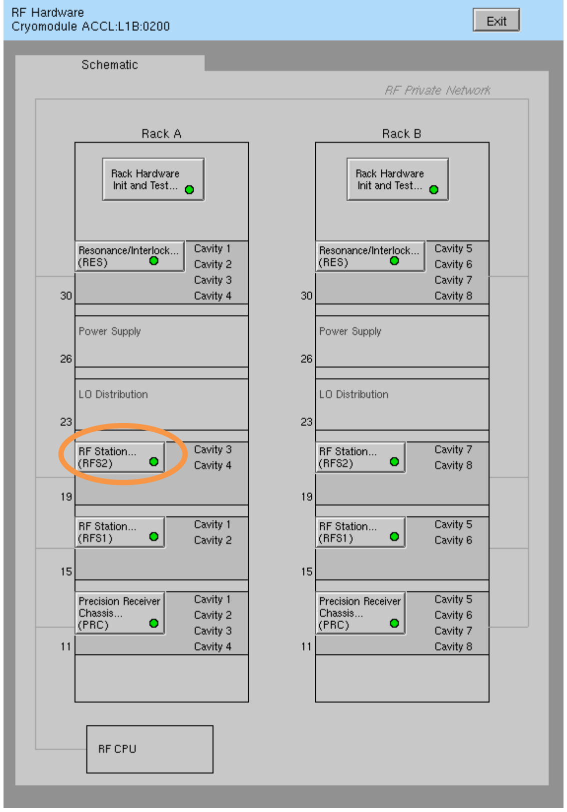
SW-FW Communication

PRODUCTION

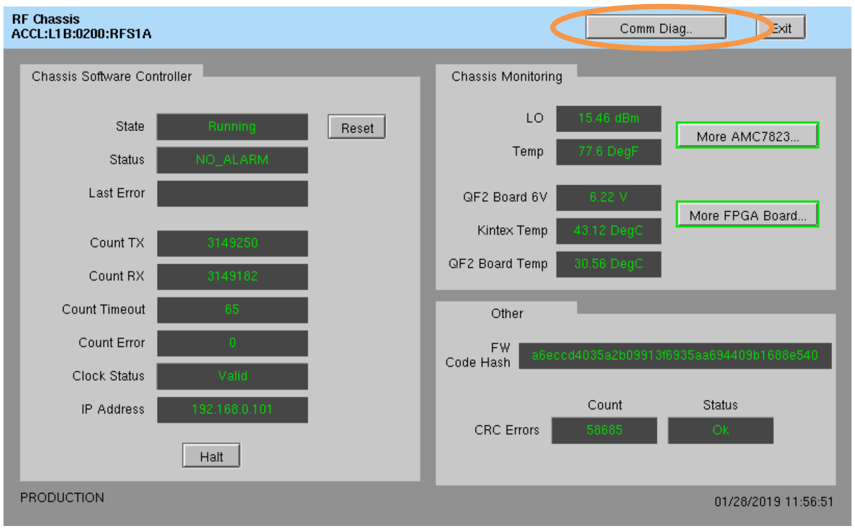
01/24/2019 16:18:48

Rack checkout display:

- Run RFS/PRC Rack Init and Test: executes wrapper script that logs into the LLRF CPU and runs lcls2_rack.sh. Click on Log… to view script output.
- Resonance/Interlock Initialize Chassis: executes wrapper script that logs into the LLRF CPU and executes res_ctl.py -a <ip> -b <bitfile> -m <file>. This Log… file contains all output from the FEED launcher (not just Res init)—so in the history, you'll see rack checkout, cavity ramp, pulse control, etc.
- *Note* that there are no automated checks of RFS<->RES communication. There is a diagnostic display for you to manually check that status:



RF Hardware
Cryomodule ACCL:L1B:0200



RF Chassis
ACCL:L1B:0200:RFS1A

Chassis Software Controller

State: Running
Status: NO_ALARM
Last Error:
Count TX: 3149250
Count RX: 3149182
Count Timeout: 65
Count Error: 0
Clock Status: Valid
IP Address: 192.168.0.101

Chassis Monitoring

LO: 15.46 dBm
Temp: 77.6 DegF
QF2 Board 6V: 6.22 V
Kintex Temp: 43.12 DegC
QF2 Board Temp: 30.56 DegC

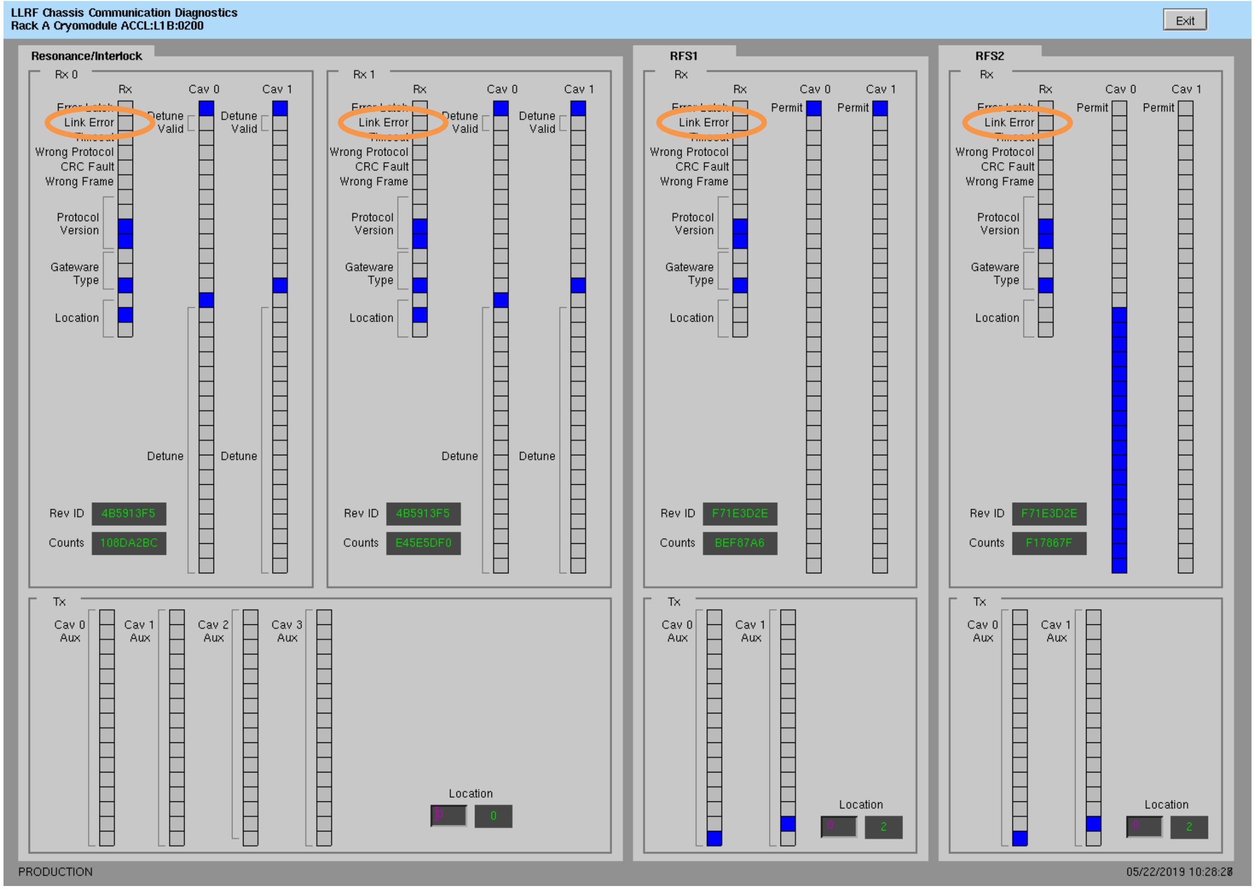
Other

FW Code Hash: a6eccd4035a2b09913f6935aa694409b1688e540

	Count	Status
CRC Errors	58685	Ok

PRODUCTION

01/28/2019 11:56:51



LLRF Chassis Communication Diagnostics
Rack A Cryomodule ACCL:L1B:0200

Resonance/Interlock

Rx 0: Rev ID 4B5913F5, Counts 108DA2BC

Rx 1: Rev ID 4B5913F5, Counts E45E5DF0

RFS1: Rev ID F71E3D2E, Counts BEF87A6

RFS2: Rev ID F71E3D2E, Counts F17867F

Tx Location 0

RFS1 Tx Location 2

RFS2 Tx Location 2

PRODUCTION

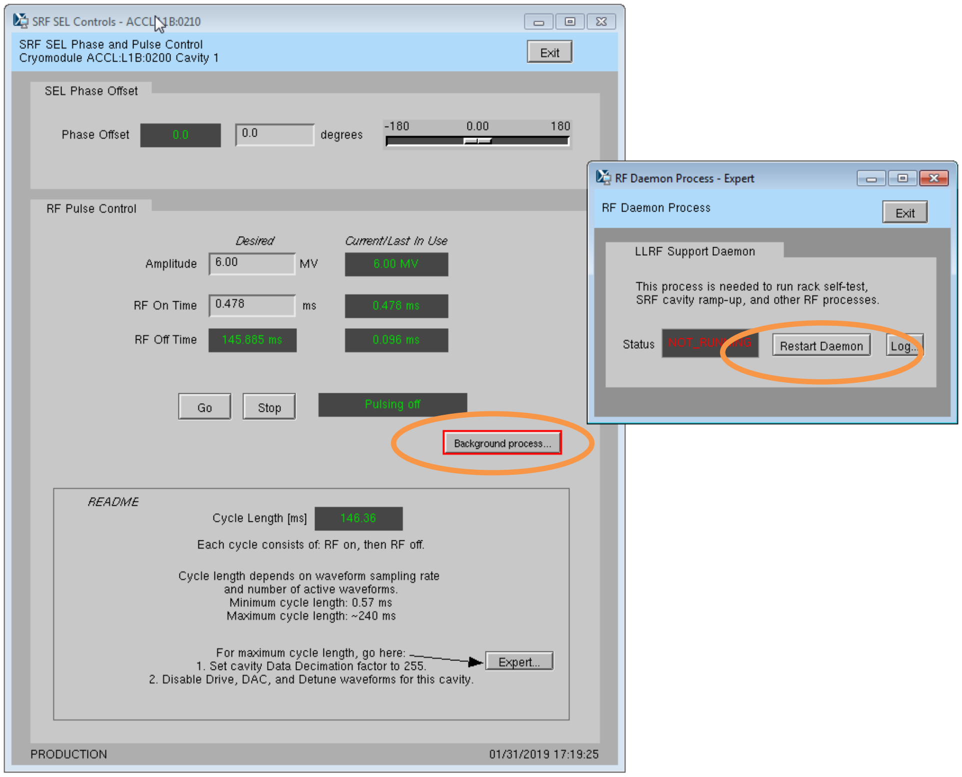
05/22/2019 10:28:28

- The top row shows the RES status; the bottom the RFS status. In the ‘Rx’ sections, the ‘Link Error’ bit is set (blue) if there is a problem. (This is a snapshot of a working system.) There is other useful data on this display too.

9. FEED ‘Launcher’

The FEED python launcher program provides the ability to use EPICS PVs to launch and provide status for external scripts. There is a configuration file that defines the shell commands it runs and their associated PV names (section 6c).

On any display that relies on the launcher, there is a red alarm if the launcher is off + a button to a display from which you can restart it. *Note* that the launcher will not start if any of the PVs it uses are offline.


SRF SEL Controls - ACCL:L1B:0210

SRF SEL Phase and Pulse Control
Cryomodule ACCL:L1B:0200 Cavity 1

SEL Phase Offset

Phase Offset: 0.0 | 0.0 degrees

RF Pulse Control

	Desired		Current/Last In Use
Amplitude	6.00	MV	6.00 MV
RF On Time	0.478	ms	0.478 ms
RF Off Time	145.885 ms		0.096 ms

Pulsing off

README

Cycle Length [ms]: 146.36

Each cycle consists of: RF on, then RF off.

Cycle length depends on waveform sampling rate
and number of active waveforms.
Minimum cycle length: 0.57 ms
Maximum cycle length: ~240 ms

For maximum cycle length, go here:
1. Set cavity Data Decimation factor to 255.
2. Disable Drive, DAC, and Detune waveforms for this cavity.

PRODUCTION — 01/31/2019 17:19:25

RF Daemon Process - Expert

RF Daemon Process

LLRF Support Daemon

This process is needed to run rack self-test,
SRF cavity ramp-up, and other RF processes.

Status: NOT_RUNNING

If you press the Restart Daemon button but are prompted for a password, your account is not authenticated to laci@lclsapp2. Ask Wesley Moore to remedy this.

Restart Daemon performs the following:

```
ssh laci@lclsapp2
/etc/init.d/st.rf_control restart
```

10. Change IP Address of FPGA board (QF2pre) or Program 'Fresh' FPGA board

Avoid two QF2pres with the same IP address on the LLRF internal network at the same time. So if you need to swap IPs between two boards, called X and Y below, you should:

i. Halt communication between EPICS and relevant chassis
ii. Disconnect X from the LLRF network
iii. Update the IP address for Y (instructions below)
iv. Disconnect Y from the LLRF network
v. Reconnect X to the LLRF network
vi. Update the IP address for X
vii. Reconnect Y to the LLRF network

If it is an unconfigured chassis, then there are probably no collisions and you could leave the other chassis connected during your work.

a. Following instructions from Section 7, halt communication between EPICS and relevant chassis
b. Log into LERF workstation or server
   (lcls01/2/3/ lclsapp1 with individual user id)

c. Log into cpu:

```
iocConsole <cpuname>
OR
ssh laci@<cpuname>
```

(If prompted for login, type 'laci' and hit enter.)

d. Change directory:

```
cd /usr/local/lcls/package/lcls2_llrf/software/submodules/qf2_pre
```

i. If it is a 'fresh' board still set to factory defaults:

```
su –
ifconfig eth1 192.168.2.31
ifconfig eth0 192.168.1.31
ping 192.168.1.127 (and verify response)
exit
python -m qf2_python.scripts.update_spartan_6_configuration -X -t 192.168.1.127 -s IPV4_UNICAST_IP=<newip>
python -m qf2_python.scripts.update_spartan_6_configuration -X -t 192.168.1.127-s IPV4_UNICAST_MAC=<mac>
python -m qf2_python.scripts.verify_spartan_6_configuration -X -t 192.168.1.127
python -m qf2_python.scripts.update_spartan_6_configuration -t 192.168.1.127 -s IPV4_UNICAST_IP=<newip>
python -m qf2_python.scripts.update_spartan_6_configuration -t 192.168.1.127 -s IPV4_UNICAST_MAC=<mac>
python -m qf2_python.scripts.verify_spartan_6_configuration -t  192.168.1.127
python -m qf2_python.scripts.update_spartan_6_configuration -X -t 192.168.1.127 -s AUTOBOOT_TO_RUNTIME=1
su –
ifconfig eth0 192.168.0.31
ifconfig eth1 192.168.1.31
exit
```

Then power-cycle chassis. Then:

```
ping <newip> (and verify response)

python -m qf2_python.scripts.verify_spartan_6_configuration -X -t <newip>
python -m qf2_python.scripts.verify_spartan_6_configuration -t <newip>
python -m qf2_python.scripts.reboot_to_runtime -t <newip>
```

ii. If it is a board previously in use and already has a 192.68.0.*** IP, an assigned MAC, and AUTOBOOT_TO_RUNTIME set:

```
python -m qf2_python.scripts.update_spartan_6_configuration -X -t 192.168.1.127 -s IPV4_UNICAST_IP=<newip>
python -m qf2_python.scripts.verify_spartan_6_configuration -X -t 192.168.1.127
python -m qf2_python.scripts.update_spartan_6_configuration -t 192.168.1.127 -s IPV4_UNICAST_IP=<newip>
python -m qf2_python.scripts.verify_spartan_6_configuration -t  192.168.1.127
```

Then power-cycle chassis. Then:

ping <newip> (and verify response)

```
python -m qf2_python.scripts.verify_spartan_6_configuration -X -t <newip>
python -m qf2_python.scripts.verify_spartan_6_configuration -t <newip>
```

e. Following instructions from Section 7, reset communication between EPICS and relevant chassis

f. Perform other checkout if desired/possible. For example, for a RFS or PRC, run prc.py or run rack checkout.

11. New Version of lcls2_llrf

Create a git tag for your new version of the lcls2_llrf repo. Typical tag names are lerf-R0-0-<revision>. Look at the tag list in git to choose your new tag name. Check out that new tagged version (section 6a):

```
cd /usr/local/lcls/package/lcls2_llrf
git pull
git checkout <tagname>
```

12. New FW Version

To use a new version of the RFS/PRC or RES firmware, you'll need to copy the bitfile to the appropriate directory (Section 6b.) and update the 'current' symbolic link in that directory to point to your new bitfile. Then run the appropriate initialization script (Section 8) to program the FPGA(s) with the new bitfile(s).

13. Expose New FW Registers in EPICS

If the new firmware has new registers that must be exposed via EPICS, you'll also need to create and install new versions of the FEED EPICS module and RF EPICS IOC application. Typically Sonya Hoobler make these changes, but in her absence, please use the contacts below.

a. Ask Carlos Serrano (cserrano@lbl.gov) or other expert to:

Create new versions of the FEED register substitutions file using the FEED leep command line interface. At the LBL or SLAC test stand run these commands (example for RFS FW):

```
cd <FEED top>/src/python
python -m leep.cli leep://<ipaddress> template --short rfs_registers_short.substitutions
python -m leep.cli leep://<ipaddress> template rfs_registers.substitutions
```

and commit the updates files to the FEED LBL gitlab repo src/Db directory:

i. For RFS/PRC, the updated files will be:
   - rfs_registers.substitutions
   - rfs_registers_short.substitutions

ii. For RES, the updated files will be:
   - res_registers.substitutions
   - res_registers_short.substitutions

Make a new FEED tag with these changes. Look at RELEASE_NOTES.SLAC to choose an appropriate new tag name. (Also please update RELEASE_NOTES.SLAC with info about the new tag.) The FEED repo is in gitlab.lbl.gov. Contact Carlos for access, if needed.

```
git checkout master
git pull
git tag –a "SLAC tag for new FW version…" <tagname>
git push origin master
git push origin tag <tagname>
```

b. Ask Garth Brown (gwbrown@slacs.stanford.edu) or Carolina Bianchini (carolina@slac.stanford.edu) to make a new tag of RF. RF is in the SLAC repo rf/RF.git.

   Modify configure/RELEASE.local: change FEED_MODULE_VERSION to the new FEED tag name. Push these changes to the git repo and make a new git tag.

c. In the LERF file system, check out the new tagged version of FEED (Section 6f) from the LBL gitlab repository. Modify the top-level Makefile to comment out these lines:

```
DIRS += feedApp
feedApp_DEPEND_DIRS = configure src
```

   Then compile by typing 'make'.

d. In the LERF file system, check out your new tagged version of RF (Section 6g) from the SLAC repository. Compile by typing 'make'. Update the 'current' symbolic link in the RF directory to point to your new version. Then reboot sioc-l1b-rf01 and sioc-l1b-rf02

14. Verify QF2-pre Network Settings

From John Jones:

I suggest you disconnect all but one board in the system then work through each board in turn, running:

```
python -m qf2_python.scripts.verify_spartan_6_image -X -t [CURRENT_IP]
```

for the bootloader settings and:

```
python -m qf2_python.scripts.verify_spartan_6_image -t [CURRENT_IP]
```

for the runtime, and make sure that:

a) The bootloader and runtime images have the same settings for IP and MAC.

b) That they are unique in the overall network.