Observing with The Rawlings Array (LOFAR-UK UK608)

LOFAR-IS-UK1-MA-0008

December 9, 2013

Abstract

A quick and simple guide to executing your scheduled observations using the Rawlings Array at Chilbolton. This guide covers both 'integrated data' collection - array co-variances, subband and beamlet statistics, along with 'raw data' observing and processing examples using the ARTEMIS backend.

Contents

1 Overview of The Rawlings Array

The Rawlings Array (LOFAR-UK UK608) is one of 8 international stations of the LOw Frequency ARray (LOFAR) currently operating. It may be used in conjunction with the other member stations of the array, or as a stand-alone telescope. This manual is intended as a 'quick-start' guide to observing with The Rawlings Array in stand-alone mode, and should be used in conjunction with the reference material available on the Observers page of the LOFAR-UK Wiki.

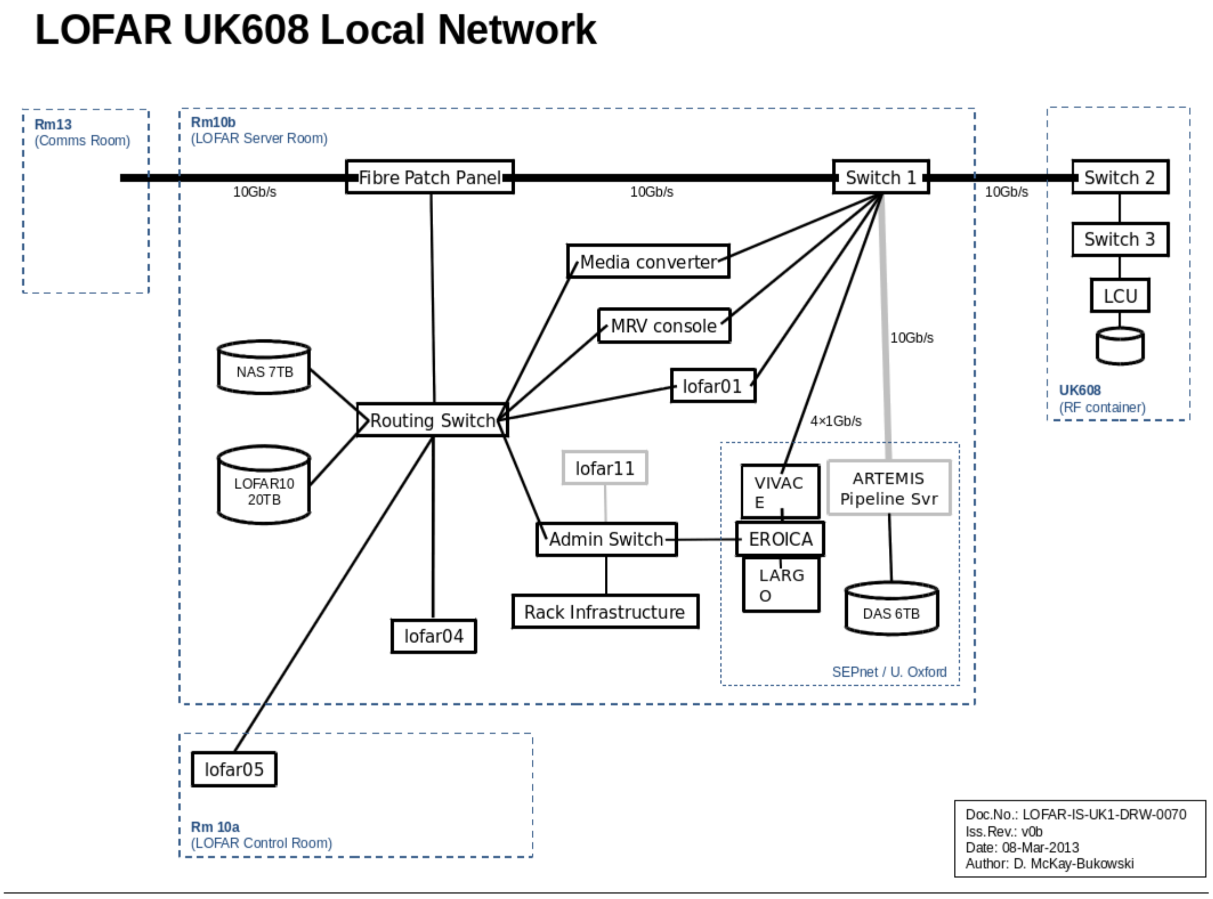

Figure 1: This figure shows the computing layout for the UK608 network. Access to the station from the outside world is through LOFAR01, from which the other units may be accessed. Statistics are recorded directly on the LCU, and later moved to to NAS_DATA for storage. Data requiring further processing via ARTEMIS flows from the station LCU to the processing units VIVACE or LARGO, and then finally to either the storage units NAS or OFFLINE_NS.

Table 1: The Names and IP addresses for the UK608 computing units.

Unit	IP address
LOFAR01	193.63.181.13
LCU	192.168.154.1
VIVACE1	192.168.1.65

2 Accessing the Station

2.1 LOFAR-UK Account Request Form

Well in advance of your scheduled observing run, copy and paste the following form (also available in a txt file on the Wiki), and email to **lofar-net@stfc.ac.uk**.

RAWLINGS ARRAY USER ACCOUNT REQUEST FORM
PLEASE FILL IN ALL FIELDS THAT APPLY
Return to lofar-net@stfc.ac.uk when complete

FULL NAME :

FRIENDLY NAME (if applicable):

EMAIL :

INSTITUTE :

PROJECT :

SUPERVISOR (if applicable):

LOFAR-UK CONTACT :

PREFERRED USER NAME :

REMOTE ACCESS LEVEL REQUIRED (default SSH):

HOST FQDN/IP/NETWORK :

2.2 Getting Started

A temporary password will be provided for lofar01 which must be changed on first login. Currently, all new accounts are given SSH access by default (web-based file management is being investigated; web-only access will be available should this be introduced). All communication must be a secure form, e.g. SSH/SCP/SFTP/HTTPS. FTP is insecure and is not supported.
As only known host computer systems are permitted access through the UK608 firewall you need to provide a fixed IP address. This will most likely be your institution (e.g. mymachine.stfc.ac.uk), remembering that most home broadband providers do not provide a static IP address by default.
When you receive your temporary password:

- Login into lofar01. Any problems with logging in should be reported to lofar-net@stfc.ac.uk.
- Change the temporary password.
- Check that you can login to the lcu as 'user6', and the acquisition machine vivace as 'observer'[1] from lofar01, without entering a password. If you find that you cannot, get in touch with lofar-net@stfc.ac.uk before proceeding further.

Check that you can successfully access all the machines necessary for your observation well in advance of your run

[1]Observers should never normally need to be logged into vivace, primarily as it is usually running at 100% CPU. All commands for this machine should be executed remotely.

3 Observing Remotely: Practicalities

Ensure that you enable x-forwarding when logging in (e.g. using ssh -Y).

In order to remain logged into the LCU (crucial when working remotely), either the use of the ‘nohup’ command, or the GNU screen tool is recommended.

3.1 Nohup

The use of ’nohup’ preceding a Unix command allows that process to continue running, even when the user has logged out. For ease of reading, the rest of this guide assumes the user will use ’nohup’ when running observations.

3.2 GNU Screen

The GNU screen tool can also be used for the same purpose. The following demonstrates the use of the GNU screen tool.

Whilst logged into the LCU, type:

```
> screen
```

and run the ’top’ command as an example. Press

```
Ctrl+a
Ctrl+d
```

which detaches the session to enable it to keep running, even if the terminal is closed/connection is lost.

```
 > screen -r
```

reattaches the screen. For more info on GNU screen, see Appendix A of this manual.

Please ensure that you kill all screen instances when the observation is completed.

4 Controlling the Station

The operation of the station is governed by a set of software levels, each containing a group of station processes which are started/stopped as appropriate to the specified software level, without manual intervention from the user.

To start, login to the LCU, and type:

```
 > swlevel
```

Before beginning observations, this level should be at 0. If it is not, get in touch with the previous observer, or lofar-net@stfc.ac.uk. Only software levels 0-3 are used in single station mode (levels 4-6 being utilised for ILT observations). Everything required for single-station observations is covered by software level 3.

Set the software level to 3:

```
 > swlevel 3
```

and wait for the prompt to return (this can take a minute or so).

If HBA use is desired:

```
 > poweruphba.sh 5
```

which powers up the HBA in RCU mode 5, otherwise, move on to the next step.

In the next few sections several examples for observing with the station are given, including taking beam-formed data, generating statistics, recording array covariances and processing observed raw data further with ARTEMIS. For more extensive details of the available observing options, please refer to the LOFAR Single Station Manual, available on the wiki.

Table 2: LOFAR Station Receiver Modes, reproduced for quick reference from the Single Station Manual. Note that RCU modes 1 and 2 are NOT available on UK608, and are hence excluded from the table. RCU mode 5 is inverted, but the poweruphba script automatically applies the inversion flag. * Note RCU mode 6 has not yet been used on UK608.

Mode	RCU Input	Frequency/MHz	Actual Passband/MHz	Receiver Band/MHz
0	-	-	-	-
3	LBH	10-90	10-80	0-100
4	LBH	30-90	30-80	0-100
5	HBA	110-190	115-185	200-100
6	HBA	170-230	170-220	160-240*
7	HBA	210-270	215-260	200-300

5 Station Tracking

In this section a step by step guide for taking beam-formed data with the station is given.

The beamctl command is used to instruct the station to take the desired beam-formed data. An example of a beam-formed observation with the HBA is:

```
> beamctl --antennaset=HBA_JOINED --rcus=0:191 --rcumode=5 --subbands=30:49\
--beamlets=81:100 --anadir=0.4,1.2,J2000 --digdir=0.4,1.2,J2000 &
```

which gives a beam covering 20 subbands, tracking a target with coordinates RA=0.4, DEC=1.2 (RAD) in J2000. **Note that a beamlet is one subband beamformed for a specific direction**. The number of subbands and beamlets for a given beamctl command must always be the same. New beams can be assigned to unused beamlet numbers without killing the existing beams so long as the other options chosen do not cause conflicts.

Common parameters include:

```
 --antennaset   Antenna set selection - choices given in /opt/lofar/etc/AntennaSets.conf
                When using HBA_DUAL option, the serdes splitter must be switched on
                (see Station Data Manual).

--rcus          Receiver selection.

--rcumode       Receiver mode selection (possible values given in Table 2).

--subbands      Subband selection, any integer between 0 and 511. Number of subbands
                must be equal to the number of beamlets.

--beamlets      Beamlet selection, any integer between 0 and 243.

--digdir        Specify longitude, latitude and type, where type is the name of the
                coordinate system used by LOFAR. Most common systems in use in
                e.g. CASA are accepted. A duration can also be specified, useful e.g.
                if several pointings are used in the same command for specified time periods.

--anadir        Specify longitude, latitude and type for the analogue HBA tile beams,
                similarly to --digdir above.

--calinfo       Print information of available station calibration tables.

--help          Print help message.
```

Note there are few software catchs for bad instructions. Things to be aware of include:

- The beamctl command should always be used where possible instead of the rspctl command.

- The –antennaset option selected must be compatable with the –rcumode. Otherwise beamctl will operate regardless and produce rubbish data.
- The frequency ranges/subbands of the observation should be adjusted such that the total number of subbands across all beams is not greater than 244.
- HBA observing only: Each HBA tile beam, using analogue beamformers, can only point in one direction at a time, and therefore all digital pointing directions must be close enough to one another to fall within the tile beam. At 150 MHz, the tile beam width is approximately 30 degrees, and is inversely proportional to the frequency (see Station Manual for more detail).
- Only one clock mode can be selected and no RCU can be in more than one mode.

Run the beamctl commands. If these are successful, a message to the effect that 'all pointings accepted' should appear. The station is now taking beam-formed data.

To change e.g. the pointing direction, a new set of beamctl commands need to be run, and the existing ones killed. To do this type:

```
> killall beamctl
```

6 Recording Statistics

Subband or beamlet statistics, and array co-variances are recorded directly on to the LCU, and transferred at the end of the observation for storage or further processing.

7 Subband Statistics

To plot the subband statistics on the LCU:

```
> rspctl --statistics=subband [--integration=<seconds> --select=<set>],
```

Optional parameters are integration which specifies the time resolution, which by default is 1s, and select, which allows specification of RCUs to be plotted (the default is all). This data may be alternatively written directly to a file on the LCU with the addition of the duration parameter.

```
> rspctl --statistics=subband --integration=2 --duration=120 --directory=/localhome/data/test
```

The above example records the subband statistics for two minutes with a time resolution of 2s. The data will be written to a set of files, one for each RCU. The file name is composed of a timestamp, an 'sst' to denote a subband statistics file , and the RCU number.

8 Beamlet Statistics

Beamlet statistics can only be written directly to the LCU. Plotting the beamlet statistics follows a similar procedure to that for the subband statistics.

```
> rspctl --statistics=beamlet --integration=2 --duration=120 --directory=/localhome/data/test
```

The above example records the beamlet statistics for two minutes with a time resolution of 2s. The data will be written to a set of files, one for each RCU. In this case file name is composed of a timestamp, an 'bst' to denote a subband statistics file , and the RCU number.

9 Array Co-variances

If you wish to record array co-variances (used for all-sky images) for all subbands, the **LOFAR Station Calibration Cookbook** provides a method to loop over all subbands and record array-covariances from each, with 1s time resolution, and **should be used** for this purpose.

Please consult with observing support before doing this for the first time, taking array co-variances involves modification of the station CalServer file, and care needs to be taken to ensure subsequent observations are not adversely affected.

As a brief summary, array co-variances, or visabilities, can only be taken for one subband at a time. The required subband can be selected with

```
> rspctl --xcsbband=<int>.
```

Similarly to the beamlet and subband statistics commands above, the array co-variances can be plotted and recorded to the LCU with:

```
> rspctl --xcstatistics [--duration=<seconds>] [--integration=<seconds>] [--directory=<directory>]
```

The resultant data file is an array co-variance cube, or 'acc' file.

10 LCU Data Storage Policy

Once your observations are completed, the data files should be transferred off the LCU to either /mnt/NAS_DATA or your home institution. This should be done immediately. **No data should be left on the LCU after your observation**.

11 ARTEMIS: Processing Raw Data

ARTEMIS, or the Advanced Radio Transient Event Monitor and Identification system, is an advanced CPU-GPU telescope backend that takes the raw voltage level beam-formed data generated by the station and performs a variety of tasks, such as channelisation of raw complex data using a polyphase filter, generation of Stokes parameters, real-time RFI excision, integration and dispersion searching for fast transients.

ARTEMIS may be used to capture raw beamformed data directly from the station, and process it further. At the moment, it is most straightforward to set up this with a series of customised scripts according to the needs of your observation. An example walk-through of the process is given here, with observers expected to customise according to their own needs.

Three machines are involved. The LCU must be set up in the normal fashion to track the source of interest. An acquisition machine (Vivace) must be instructed to listen for the raw data being broadcast from the LCU via UDP packets, and perform the necessary post-processing. Finally, both these processes may be controlled from LOFAR01 with an overarching script.

11.1 The LCU: Station Tracking

The station should be set to observe your source as outlined in Section *. This can be done remotely by specifying the desired beamctl commands as a bash script and storing this in your home directory, as, e.g. call_observe.csh.

11.2 Acquisition (Vivace): XML Configuration Files

Once the station is taking data, the processing (acquisition) units need to be instructed to listen for, process if required, and record the station data. These instructions are given using a set of user specified XML configuration files.

There is one server script, and two data stream scripts (the acquisition machine vivace has two CPUs, and this setup optimises use of the machine). The server script tells vivace to listen for the data from the LCU, and to split it into two data streams. The data stream scripts then contain instructions for vivace as to whether any additional processing is required (e.g. RFI removal), what parts of the data to write (e.g. polarisations) and finally where to write the data to.

The following set of example XML configuration files can be found under:

```
/mnt/nas_data/Observer_Example/config on LOFAR01.
```

and are also downloadable from the LOFAR-UK Wiki. These files should be prepared in advance for taking beam-formed data.

11.3 Acquisition (Vivace): Recording and Processing Raw Station Data

The next step is to leave these LCU processes running, and instruct the processing units to capture the generated data from the station. *You must remain logged into the LCU, otherwise these processes will terminate.*

The following example set of commands, 'observe_vivace.sh' instruct vivace to record the beam-formed data from the LCU, split into two datastreams for both CPUs. These utilise the example xml files given on the wiki, which include self-explanatory options such as the polarisations to record, and whether any basic RFI removal is to be done. These may be used as a starting template for your own observations. The 'sleep 30' commands signifies how long the datastreams are recorded (i.e. the length of your observation), and the final command calls a script to kill the recording.

```
cpu1 udpBFPipelineStream1 --config=/data/Observer_Example/config/stream1_t311.xm
l >& /data/RAW_DATA/log1.dat &
cpu2 udpBFPipelineStream2 --config=/data/Observer_Example/config/stream2_t341.xm
l >& /data/RAW_DATA/log2.dat &
sleep 5
cpu0 serverMain --config=/data/Observer_Example/config/server2.xml >& /data/RAW_
DATA/log0.dat &
sleep 30
/data/Observer_Example/scripts/killobs
```

11.4 Running the Observation on ARTEMIS

Once you have set up your scripts on the LCU and Vivace, with the correct xml configuration files, you are ready to begin the observation. This can all be done from LOFAR01 with a single overarching script.

```
 #! /bin/bash
ssh lcu "/data/home/user6/Scripts/call_observe.csh" &
sleep 30
echo "ON SOURCE"
ssh artemis@vivace1 "sh /data/Commissioning/NewObs/observe_vivace1.sh" &
pidvivace1=$!
wait $pidvivace1
exit
```

11.5 Data Storage Policy

As per the LCU, data should not be stored on the aquisition machines, but instead should be moved to the storage disk (NAS_DATA) or your home institution. Note that the storage disk fills up quickly, so data should not be stored long term here.

12 Finishing Observing

When you have finished observing with the station, set the software level on the LCU back to 0:

```
> killall beamctl
> swlevel 0
```

Once you have finished your observations, please ensure you complete the following:

- Move your data off the LCU and/or aquisition machines in line with the data storage policy.
- Kill all GNU Screen sessions
- Email the next observer with station hand-over details.
- Feedback to observing support any difficulties with observing.

13 Transient Buffer Boards

Transient buffer Boards are not currently in use on The Rawlings Array. It may be possible for this to be used, but should be treated as a commissioning task. Please refer to central LOFAR documentation on TBBs, and contact the observing support team for further information if required.

14 357 Mode

357 mode has not been fully tested on The Rawlings Array. If you are interested in commissioning this mode, please refer to the technical note on 357 mode observing with KAIRA on how to get started (available on the LOFAR-UK Wiki).

15 8/16 Bit Mode

16 bit mode is currently used as standard on The Rawlings Array. 8 bit mode usage would be treated as a commissioning task - please contact observing support for further details. See also `http://www.lofar.org/operations/doku.php?id=singlestation:start#making_observations_with_4-_and_8-bit_sample_mode`

16 Useful Contacts & References

16.1 Access Difficulties

Any problems with station logins or environment difficulties should be addressed to lofar-net@stfc.ac.uk.

16.2 General Observing Support

There is a comprehensive set of general information relating to single station use (not Chilbolton specific) on the LOFAR Wiki here: `http://www.lofar.org/operations/doku.php?id=singlestation:start&s[]=single&s[]=station`

Questions regarding observing not covered here should be addressed to:

- lofar-obs@stfc.ac.uk

16.3 LOFAR-UK

Any queries regarding LOFAR-UK should be addressed to:

16.4 Useful References

The most recent versions of all reference material referred to in this manual are available on the LOFAR-UK Wiki.
LOFAR Station Manual
SEPIA Documentation
LOFAR Station Calibration Manual
357 Mode Observations with KAIRA

16.5 Acknowledgements

This document was largely assembled by Louise Ker, with thanks to Aris Karastergiou, Philp Best, Derek McKay, and Alan Doo.

17 Appendix A: Example XML Configuration Files

This is the example ARTEMIS PELICAN XML script server2.xml, as used in Section 11.3. This script splits the raw data coming from the LCU into two streams by subband number, along with some details of the observation, e.g. number of polarisations, clock value etc.

```
 <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE pelican>

<configuration version="1.0">

  <server>

    <chunkers>
      <LofarDataSplittingChunker>
        <connection host="127.0.0.1" port="4347"/> <!--Change-->
        <!--  -->
        <subbandsPerPacket value="61" /> <!-- 31 or 61 or 62 -->
        <import file="/data/Observer_Example/config/c64i16_common.xml"/>
        <!--  -->
        <Stream1 subbandStart="0" subbandEnd="30"/>
        <Stream2 subbandStart="31" subbandEnd="60"/>
        <data type="LofarTimeStream1" />
        <data type="LofarTimeStream2" />
      </LofarDataSplittingChunker>
    </chunkers>

    <buffers>
      <LofarTimeStream1>
        <buffer maxSize="800000000" maxChunkSize="50000000"/>
      </LofarTimeStream1>

      <LofarTimeStream2>
        <buffer maxSize="800000000" maxChunkSize="50000000"/>
      </LofarTimeStream2>
    </buffers>

  </server>
</configuration>
```

and configuration file c64i16_common.xml as called by the above script.

```
 <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE pelican>

<configuration version="1.0">

    <samplesPerPacket value="16" />
    <nRawPolarisations value="2" />
    <dataBitSize value="16" />
    <totalComplexSubbands value="512" />

    <clock value="200" /> <!-- Could also be 160 -->
    <outputChannelsPerSubband value="64" />
    <udpPacketsPerIteration value="128" />
    <integrateTimeBins value="32" />

</configuration>
```

This is the example ARTEMIS PELICAN XML script stream1_t311.xml, as used in Section 11.3. This script takes the first data stream, and performs several processing actions, e.g. PPF channelisation, RFI removal, integration.

```
 <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE pelican>

<configuration version="1.0">

  <pipeline>

    <clients>
      <PelicanServerClient>
        <server host="127.0.0.1" port="2000"/>
        <!--server host="192.168.1.65" port="2000"/-->
        <data type="LofarTimeStream1" adapter="AdapterTimeSeriesDataSet"/>
      </PelicanServerClient>
    </clients>

    <adapters>
      <AdapterTimeSeriesDataSet>
        <subbandsPerPacket value="31"/> <!-- 31 or 61 or 62 -->
        <import file="/data/Observer_Example/config/c64i32_common.xml"/>
        <!--   -->
        <fixedSizePackets value="false" />
      </AdapterTimeSeriesDataSet>
    </adapters>

    <modules>
      <PPFChanneliser>
        <import file="/data/Observer_Example/config/c64i32_common.xml"/>
        <!--   -->
        <processingThreads value="6" />
        <filter nTaps="8" filterWindow="kaiser"/>
      </PPFChanneliser>

      <StokesGenerator>
      </StokesGenerator>

      <RFI_Clipper active="false" channelRejectionRMS="3.5"
                   spectrumRejectionRMS="6.0">
<zeroDMing active="true" />
<BandPassData file="/data/Commissioning/Brenda/band220-250.bp" />
<Band matching="true" />
<History maximum="10000" />
      </RFI_Clipper>

      <StokesIntegrator>
        <import file="/data/Observer_Example/config/c64i32_common.xml"/>
      </StokesIntegrator>

    </modules>

    <output>

      <dataStreams>
        <!-- <stream name="SpectrumDataSetStokes" listeners="PelicanTCPBlobServe
r" />  -->
<!--stream name="LofarTimeStream1" listeners="DataBlobFile" /> -->
```

```
        <stream name="SpectrumDataSetStokes" listeners="SigprocStokesWriter"/>
      </dataStreams>

      <streamers>
        <SigprocStokesWriter active="true" writeHeader="true">
          <import file="/data/Observer_Example/config/c64i32_common.xml"/>
          <dataBits value="32" />
      <scale max="100" min="0" />
          <topSubbandIndex value="341"/>
          <LBA_0_or_HBA_1 value="1" />
          <subbandsPerPacket value="31"/>
          <!-- -->
          <file filepath="/data/RAW_DATA/t311" /><!--Change-->
          <params telescope="LOFAR" nPolsToWrite="1"/><!--Change-->
          <RAJ value="032100.0" />
          <DecJ value="670000.0"/>
          <TelescopeID value="3"/>
          <MachineID value="10"/>
        </SigprocStokesWriter>
      </streamers>

    </output>

  </pipeline>
</configuration>
```

This is the example ARTEMIS PELICAN XML script stream2_t341, as used in Section 11.3. This script takes the second data stream, and performs the same several processing actions, e.g. PPF channelisation, RFI removal, integration.

```
 <?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE pelican>

<configuration version="1.0">

  <pipeline>

    <clients>
      <PelicanServerClient>
        <server host="127.0.0.1" port="2000"/>
        <!--server host="192.168.1.65" port="2000"/-->
        <data type="LofarTimeStream2" adapter="AdapterTimeSeriesDataSet"/>
      </PelicanServerClient>
    </clients>

    <adapters>
      <AdapterTimeSeriesDataSet>
        <subbandsPerPacket value="30"/> <!-- 31 or 61 or 62 -->
        <import file="/data/Observer_Example/config/c64i32_common.xml"/>
        <!--   -->
        <fixedSizePackets value="false" />
      </AdapterTimeSeriesDataSet>
    </adapters>

    <modules>
      <PPFChanneliser>
        <import file="/data/Observer_Example/config/c64i32_common.xml"/>
        <!--   -->
        <processingThreads value="6" />
        <filter nTaps="8" filterWindow="kaiser"/>
      </PPFChanneliser>

      <StokesGenerator>
      </StokesGenerator>

      <RFI_Clipper active="false" channelRejectionRMS="3.5"
                   spectrumRejectionRMS="6.0">
<zeroDMing active="true" />
<BandPassData file="/data/Commissioning/Brenda/band220-250.bp" />
<Band matching="true" />
<History maximum="10000" />
      </RFI_Clipper>

      <StokesIntegrator>
        <import file="/data/Observer_Example/config/c64i32_common.xml"/>
      </StokesIntegrator>

    </modules>

    <output>

      <dataStreams>
        <!-- <stream name="SpectrumDataSetStokes" listeners="PelicanTCPBlobServe
r" />  -->
<!--stream name="LofarTimeStream1" listeners="DataBlobFile" /> -->
```

```
        <stream name="SpectrumDataSetStokes" listeners="SigprocStokesWriter"/>
      </dataStreams>

      <streamers>
        <SigprocStokesWriter active="true" writeHeader="true">
          <import file="/data/Observer_Example/config/c64i32_common.xml"/>
          <dataBits value="32" />
      <scale max="100" min="0" />
          <topSubbandIndex value="341"/>
          <LBA_0_or_HBA_1 value="1" />
          <subbandsPerPacket value="30"/>
          <!-- -->
          <file filepath="/data/RAW_DATA/t341" /><!--Change-->
          <params telescope="LOFAR" nPolsToWrite="1"/><!--Change-->
          <RAJ value="032100.0" />
          <DecJ value="670000.0"/>
          <TelescopeID value="3"/>
          <MachineID value="10"/>
        </SigprocStokesWriter>
      </streamers>

    </output>

  </pipeline>
</configuration>
```

18 Appendix B: GNU Screen

For reference, the GNU screen section from the LOFAR Imaging Cookbook (v12.0, Roberto Pizzo) is reproduced here.

For some long running processes, it can be handy to have a session running on one of the units, and then leave that running in the background. This can be done by putting the task in the background and use a command like disown. Sometimes, however, it is more convenient to have a program running the in the foreground. In such a case, it is difficult to log out without killing the process. GNU screen to the rescue. GNU screen (short: screen) is a utility that creates a virtual terminal that stands on its own, and can be put in the background or foreground at will. Multiple terminals ("tabs") are also possible. A disadvantage of GCN screen is that it will not use with X-windows (forwarding) and scrolling back for previous output does not always work. For a long-during command line task, however, it is very useful. Have a first look at screen using the help option:

```
 screen -help
Use: screen [-opts] [cmd [args]]
or: screen -r [host.tty]
Options:
-a Force all capabilities into each window's termcap.
-A -[r|R] Adapt all windows to the new display width & height.
-c file Read configuration file instead of '.screenrc'.
-d (-r) Detach the elsewhere running screen (and reattach here).
-dmS name Start as daemon: Screen session in detached mode.
-D (-r) Detach and logout remote (and reattach here).
-D -RR Do whatever is needed to get a screen session.
-e xy Change command characters.
-f Flow control on, -fn = off, -fa = auto.
-h lines Set the size of the scrollback history buffer.
-i Interrupt output sooner when flow control is on.
-l Login mode on (update /var/run/utmp), -ln = off.
-list or -ls. Do nothing, just list our SockDir.
-L Turn on output logging.
-m ignore $STY variable, do create a new screen session.
-O Choose optimal output rather than exact vt100 emulation.
-p window Preselect the named window if it exists.
-q Quiet startup. Exits with non-zero return code if unsuccessful.
-r Reattach to a detached screen process.
-R Reattach if possible, otherwise start a new session.
-s shell Shell to execute rather than $SHELL.
-S sockname Name this session <pid>.sockname instead of <pid>.<tty>.<host>.
-t title Set title. (window's name).
-T term Use term as $TERM for windows, rather than "screen".
-U Tell screen to use UTF-8 encoding.
-v Print "Screen version 4.00.03 (FAU) 23-Oct-06".
-wipe Do nothing, just clean up SockDir.
-x Attach to a not detached screen. (Multi display mode).
-X Execute <cmd> as a screen command in the specified session.
```

The detach and attach commands are the 'background' and 'foreground' commands. Now start screen for real (from a frontend or compute node):

```
 > screen
```

You will get a new terminal, but otherwise everything looks the same. You can exit this 'new' terminal using exit or control-D (depending on your shell). It is more fun, however, to detach the screen session, while running a command in the foreground. So simply execute sleep for half a minute:

```
 > sleep 30
```

and detach screen, by typing control-a control-d. All screen commands start with a special control key, which by default is control-a. You should now have been returned to your previous terminal. Now list the available screen instances:

```
> screen -list
There is a screen on:
8090.pts-64.lfe001 (Detached)
1 Socket in /var/run/screen/S-rol.
```

This shows you which screen(s) are running. You can have multiple screens, although that may be confusing. Now re-attach to this screen:

```
screen -r
```

You will get back to the screen terminal, which may still be running sleep (or it might just have ended). Now create a new 'tab' inside screen. Use control-a control-c (c for 'create'). You end up in the new tab. If you want to switch to the previous tab, use control-a 0; the numbers 0 to 9 specify a tab. You can also use control-a " to get a list of tabs, and use the arrows keys & enter to select the tab (in case you have more than ten tabs). Lastly, to toggle back and forth between two tabs, you can simply type control-a control-a. In case you accidentally loose your internet connection, your session to your the LOFAR cluster will quit. 'Screen', however, will still be running. If you login again to the machine where you started screen, you can see it:

```
> screen -list
There is a screen on:
8090.pts-64.lfe001 (Attached)
1 Socket in /var/run/screen/S-rol.
```

Note that it says Attached; you can't simply attached to it using screen -r, you first have the detach the (now defunct) old screen session, and then reattach to it:

```
screen -d -r
```

Inside screen, there are a lot more commands that you can use. Type control-a ? to get a help view (space or enter to exit). The above sample session should keep you going for most tasks anyway. Note that you can easily ssh to machines from within screen (just like normal), although forwarding X ('-Y' option) won't work. So starting screen from the frontend node and then logging in to one or more compute nodes to run NDPPP, BBS and/or mwimager can work nicely. For people who don't like the the control-a shortcut, you can redefine this key in a file called .screenrc (in your home directory. Enter the following line:

```
escape ^Jj
```

If you have problems with the delete key, try this in .screenrc:

```
bindkey -k kD
bindkey -d -k kD
```

There are many other options; see for example http://www.emacswiki.org/emacs/GnuScreen. In particular, if you want to dump out what is in screen's buffer (not only what you see on the screen, but also whatever is in its scrollback buffer), you can:

- press Control-a
- type ":hardcopy -h output.txt" (output.txt can be whatever you want)

Then you can open output.txt with a text editor to have a look at what screen had in its memory. To make screen remember more lines, you can edit /.screenrc so that the line containing "defscrollback" says something like

```
defscrollback 10000
```