Station Data Cookbook

Verified:			
Name	Signature	Date	Rev.nr.
Menno Norden	.....	.....	1.2

Accepted:		
Work Package Manager	System Engineering Manager	Program Manager
Menno Norden ..... *date:*	Andre Gunst ..... *date:*	Rene Vermeulen ..... *date:*

©ASTRON 2018
All rights are reserved. Reproduction in whole or in part is prohibited without the written consent of the copyright owner.

Distribution list:

Group:	For Information:

Document revision:

Revision	Date	Section	Page(s)	Modification
0.1	2011-Oct-04	-		Creation
0.2	2012-Mar-26	-		Several corrections to the text
1.0	2012-Apr-13	-		Final corrections to version 1.0
1.1	2012-May-18	2.3,4.3,5.3,A.3		”antenna cross-correlation” to ”array covariance”
		5.3		Number of samples in a file
		3.2		Added `--rcus` to `beamctl`
1.2	2018-Jan-26	3.3,4.2,5.2,A.2		new rspctl commands –mode, –bitmode
		A.3,B		244/488 beamlets, consistency corrections

Abstract

This document provides a description of how to locally control a single LOFAR station. It provides a short overall description of the station hardware, and instructions for controlling the station-level signal processing. Detailed descriptions of data output formats are given.

Contents

1 Introduction **7**

2 Data processing at the station **7**

2.1 Antennas 7

2.1.1 Low-band antenna (LBA) 7

2.1.2 High-band antenna (HBA) 8

2.2 Receiver Unit (RCU) 9

2.3 Remote Station Processing board (RSP) 10

2.4 Transient Buffer Board (TBB) 12

2.5 Local Control Unit (LCU) 12

3 Controlling the station **12**

3.1 Software levels 12

3.2 Beam control 13

3.2.1 Coordinate systems 16

3.3 RSP control 17

3.4 TBB control 20

4 Data recording and transmission **21**

4.1 Recording subband statistics 21

4.2 Recording beamlet statistics 22

4.3 Recording array covariances 23

4.4 Recording data with TBBs 23

4.5 TBB data CEP output 27

4.6 Beamlet data CEP output 27

5 Data output formats **28**

5.1 Subband statistics file format 28

5.2 Beamlet statistics file format 28

5.3 Array covariance file format 29

5.4 TBB data file format 30

5.5 TBB CEP data packet format 31

5.6 Beamlet CEP data packet format 31

A LOFAR station command reference **35**

A.1 swlevel 35

A.2 beamctl 35

A.3 rspctl 38

A.4 tbbctl 43

B LOFAR station configuration files **46**

C CEP data recording with Wireshark **48**

C.1 Connecting a receiving computer 48

C.2 Wireshark GUI 49

C.3 tshark 50

C.4 Libcap file format 51

©ASTRON 2018

Terms and abbreviations

Table 1: Terms and abbreviations used in this document. The terminology follows definitions in [8].

Term or abbreviation	Meaning
Antenna array	An array of antenna elements
Antenna element	A device that receives radio signals at two orthogonal linear polarisations
Antenna set	A subset of an antenna array
Beam	A set of one or more beamlets with same pointing direction
Beamlet	Beamformed subband
CEP	Central Processor
FIFO	First In, First Out
FM	Frequency Modulation
HB	High Band
HBA	High-Band Antenna
LB	Low Band
LBA	Low-Band Antenna
LBH	Low Band High
LBL	Low Band Low
LCU	Local Control Unit
LOFAR	Low Frequency Array
NIC	Network Interface Controller
Polyphase filter	An FFT-based implementation of a digital band pass filter bank
PVSS	ProzessVisualisierungs und SteuerungsSystem
RCU	Receiver Unit
Receiver band	Frequency range of the Nyquist zone that is nominally being received
Receiver passband	Passband of the analogue filter that selects the received Nyquist zone
RFI	Radio Frequency Interference
RSP	Remote Station Processing
SERDES	SERialiser / DESerialiser
Slice	A set of subband (beamlet) samples nominally received at same time
Subband	Output of one passband of the polyphase filter
TBB	Transient Buffer Board
TDS	Time Distribution System

1 Introduction

This document provides instructions that should allow the reader to locally control signal processing of a single LOFAR station, as well as to record several kinds of data products. Commands and control files relevant for a single-station user are introduced in detail. The document also provides detailed output data file and ethernet frame formats. This document concentrates on hardware, software, and data formats that belong to standard LOFAR station installations, additional components possibly installed in individual stations are not discussed. However, the document is intended to provide sufficient information for users interested in developing e.g. additional data recording systems. The reader is assumed to have basic knowledge of radio signal processing and linux operating system, but previous knowledge of the LOFAR system is not strictly necessary.

2 Data processing at the station

2.1 Antennas

LOFAR station is a wide-band radio receiver, which operates at the frequency range from 10 MHz to 240 MHz. The full reception band is divided into low band (10 MHz – 90 MHz) and high band (110 MHz – 240 MHz). The frequencies in between low band and high band are not used due to RFI from FM radio transmitters.

The station signal processing has been originally designed assuming that three different antenna arrays would be used. These arrays would have been the low-band-low (LBL) antenna array, the low-band-high (LBH) antenna array, and the high-band antenna (HBA) array. However, the present standard stations do not have the LBL antennas, and the LBH antenna array is thus referred to as the low-band antenna (LBA) array.

Antennas used in LOFAR stations are briefly introduced in following two sections. Notice that the antenna array configurations are configurations of full antenna arrays. It is always possible to manually select only a subset of a full antenna array to be used e.g. in beamforming. Positions of individual antenna elements and antenna array definitions of a station are stored in station configuration files (see Appendix B).

2.1.1 Low-band antenna (LBA)

The low-band antenna array is used for reception at frequencies from 10 MHz to 90 MHz. A low-band antenna element consists of two inverted-V shaped dipole elements, which are used for receiving two orthogonal linear polarisations. The physical installation makes use of a central pole, which is planted on

Figure 1: LBA elements (left) and HBA tiles (right) on field. ©Menno Norden

the centre of a square metallic mesh ground plane. Centres of the dipoles are attached to the top of the pole, and their ends are anchored to the ground close to corners of the ground plane. Low-noise amplifiers for the two polarisations are installed inside the central pole. The amplified signals are transferred to receivers in coaxial cables.[16] Power is delivered to the antenna elements along the same coaxial cables that are used for the signal transfer.

A standard LOFAR station has 96 low-band antenna elements, which are installed in form of "randomised concentric rings" with two outlier elements [17]. In international stations all 96 antenna elements are used as a single antenna array, whereas in core stations and remote stations the LBA array is divided into two subsets of 48 antenna elements, called LBA inner and LBA outer. If both orthogonal linear polarisations are being received, all antenna elements of the inner and the outer LBA arrays cannot be used simultaneously. However, a special receiver configurations allows single linear polarisation to be received with all 96 LBA elements of a core or a remote station. It is also possible to use a sparse array configuration, where half of the antenna elements in LBA inner array are combined with half of the elements in LBA outer array.

2.1.2 High-band antenna (HBA)

The high-band antenna array is used for reception at frequencies from 110 MHz to 240 MHz. Custom design bow-tie antenna elements are used in order to facilitate design requirements for antenna element impedance and radiation pattern [2, 1]. Each antenna element receives two orthogonal linear polarisations. The antenna elements are installed inside supporting polystyrene structures in form of a regular 4×4 array with 1.25 m inter-element distances. The supporting structure is a square 5×5 m "tile", which houses 16 individual antenna elements. The tiles are installed 0.15 m apart from each other. A metallic mesh ground plane is installed on the bottom of the tile. Each tile contains low-noise amplifiers and analogue beamformers, which combine the signals from the 16 antenna elements into two "tile beams" with orthogonal linear polarisations. The analogue tile beam signals are transferred to receivers in coaxial

cables.

Because the analogue beamformers inside the HBA tiles need to be controlled, a system for transferring control data to and from the tiles is necessary. Power consumption of the 16 antenna elements in a tile is also significantly higher than that of an individual LBA antenna element. Both the control connection and the power supply are arranged using the same coaxial cables that are used for transferring the received signals. The X-polarisation cable carries 48 V power supply for the tiles, and the Y-polarisation cable carries the control signal [12].

A core station has 48 HBA tiles, which are organised as two separate antenna arrays, HBA0 and HBA1, consisting of 24 tiles each. The two arrays can be either used separately, or they can be tied together into an array of 48 tiles. A remote station has 48 tiles as well, but they are organised as a single antenna array of 48 tiles. An international station has a single antenna array of 96 tiles.

2.2 Receiver Unit (RCU)

The analogue signals from LBA elements and HBA tiles are transferred in coaxial cables to a LOFAR station cabinet, where each of the cables is connected to a receiver unit. A RCU performs input selection, followed by amplification and filtering of the analogue input signal. The conditioned analog signal is sampled with a 12-bit A/D converter. The A/D converter produces real signal samples at either 200 MHz or 160 MHz sampling frequency.[13] Combinations of filter passband and sample clock frequency are selected so that the selected frequency band always aliases around zero frequency, without frequency aliasing inside the passband. With one of the available combinations the signal spectrum will be inverted, but it can be inverted back in subsequent processing steps. Available receiver modes are listed in Table 7 on page 37.

The receiver units are numbered starting from zero. X-polarisation cables are connected to even-numbered RCUs and Y-polarisation cables are connected to odd-numbered ones [12]. The only deviation from this rule is the LBA outer field, whose X-polarisation cables are connected to odd-numbered RCUs and Y-polarisation cables to the even-numbered ones. A receiver unit has three inputs: LBL, LBH, and HBA. The LBL and LBH connectors have an 8 V bias voltage. The HBA X-polarisation connectors have the 48 V bias voltage for power supply. The HBA Y-polarisation connectors have a 3.3 V bias voltage only when communicating with the HBA tiles.

A core station or a remote station with 96 LBA elements and 48 HBA tiles has 96 RCUs. The LBA inner array is connected to the LBH inputs and the LBA outer array to the LBL inputs. Despite the naming convention in RCUs, both LBA inner and LBA outer use the same kind of antenna elements. The HBA tiles are connected to the HBA inputs.

Due to their larger number of HBA tiles, international stations have 192 RCUs. The 96 LBA elements of an international station are connected to the LBH inputs of the RCUs, and the 96 HBA tiles are

connected to the HBA inputs. In a basic installation of an international station the LBL inputs are left empty, thus facilitating later installation of an additional antenna array.

2.3 Remote Station Processing board (RSP)

After frequency band selection and A/D conversion in the RCUs, the received signals are transferred to remote station processing boards. The RSP boards perform all digital signal processing that is done at the station, and send data to central processor (CEP).

The discrete signal samples arriving from a RCU to a RSP board are first buffered in a FIFO buffer. In order to compensate for differences in signal delays in the coaxial cables, buffer length can be adjusted independently for each individual RCU. The buffer is followed by a polyphase filter, which divides the wide band input signal into so-called subbands.

A polyphase filter is a novel FFT-based implementation of a bandpass filter bank [e.g. 5], which divides the real input signal into 1024 complex subband signals. Because the real input contains two identical (mirrored) copies of the signal spectrum – one at positive and the other at negative frequencies – the lowest 512 subbands are complex conjugates of the 512 highest ones. The whole receiver passband is thus covered by the 512 lowest subbands, which are used in further processing. Depending on sample clock frequency, subband width is either 200 MHz/1024 = 195.3125 kHz, or 160 MHz/1024 = 156.250 kHz. [9]

After the polyphase filter the signal bandwidth is small enough to facilitate phased-array beamforming[1], which produces beamformed subband signals called beamlets. Although 512 subbands are produced in the polyphase filter, only 244 beamlets in 16 bit mode, and 488 beamlets in 8 bit mode can be formed by the station signal processing. Subbands used in beamforming are selected immediately after the polyphase filter [15]. The beamforming is performed in a ring, where a process handling signal from one RCU receives a partial beamforming result, adds the contribution from its own RCU, and passes the result forward to the process handling signal from the next RCU. In order to facilitate the beamforming, the RSP boards of a LOFAR station are connected together as a ring. After a full cycle in the ring, the beamformed sample is complete and ready to be transmitted to central processing.

The beamforming ring is divided into four separate lanes, each of which has its start and end point in a different RSP board. Beamformed data are output from those RSP boards that are endpoints of the beamforming lanes. In core stations the beamforming lanes can be split into two halves, allowing independent beamforming with the HBA0 and HBA1 arrays. Because 244/488 beamlets can be formed with both arrays, the splitting allows core stations to form 488/976 beamlets simultaneously, but each of them with only 24 HBA tiles. Four additional output RSP boards are defined in the split mode, i.e. data are being output from eight RSP boards simultaneously.

[1] Phased-array beam-forming approximates time-delays as phase-shifts. The approximation is reasonably accurate only if the signal spectrum is much narrower than inverse of the time-delay that is being approximated.

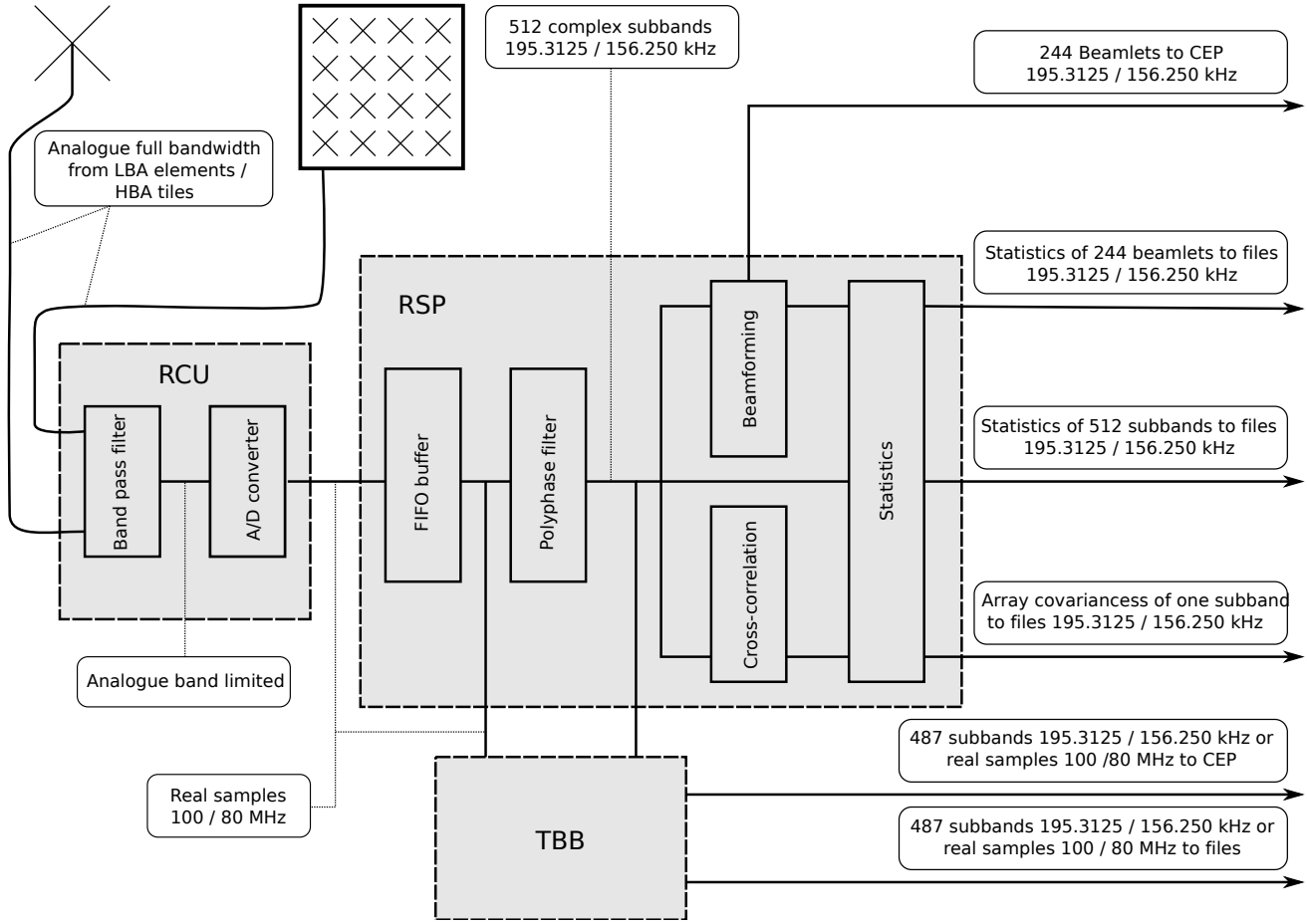

Figure 2: Functional description of data flow inside a LOFAR station. The analogue full bandwidth signal arriving from a HBA tile to a RCU is the tile beam signal produced by the analogue tile beamformer. The band widths for real samples are widths of Nyquist zones, i.e. the sample rate is twice the band width given in the figure. The subband widths are for a single subband. Notice that the functional description of this figure does not represent the exact physical data paths inside the digital signal processing system. For details of the signal processing, see [10] and [15].

In addition to the beamforming, the RSP boards can calculate so-called subband statistics (see Section 4.1), beamlet statistics (4.2), and array covariances (crosslet statistics) (4.3). Likewise the beamlets, the covariances are also formed in the ring of RSP boards, and their calculation is divided into four lanes. All these crosslet lanes start from and end to the same RSP board, which also outputs the data. The crosslet output board is usually different from the beamlet output boards. The RSP boards can also send either raw A/D converter samples or subband data to ring buffers in transient buffer boards (see Sections 2.4, 4.4).

An RSP board has four antenna processors (AP). Each AP processes two orthogonal polarisations from its connected antenna. Each RSP board thus has 8 digital inputs for the signals from the RCUs, which

defines the number of RSP boards in a station: core stations and remote stations have 12 RSP boards, whereas international stations have 24 RSP boards.

2.4 Transient Buffer Board (TBB)

Transient buffer boards are not part of the main signal processing chain, but their function is to store a short period of raw voltage data in a ring buffer. Writing to the buffer can be stopped when a triggering event is detected, after which the data can be read from the buffer (see Sections 3.4, 4.4, 4.5).

TBBs are physically connected to the RSP boards, which provide the raw data that is buffered to the boards. Two kinds of data can be buffered: in "transient" mode real signal samples are copied to TBBs immediately after the FIFO buffers in RSP boards, whereas in "subbands" mode complex subband data produced by the polyphase filter are copied to the TBBs. TBB memory is divided into pages, which are large enough for 1024 12-bit data samples in "transient" mode. Because the subband data is in 16-bit format, only 487 of the 512 subbands can be recorded in "subbands" mode. [14]

One TBB is capable of recording data from 16 RCUs, and each TBB is thus connected to two RSP boards. Core stations and remote stations thus have 6 TBBs, whereas international stations have 12 TBBs.

2.5 Local Control Unit (LCU)

Local control unit is a server computer running redhat linux operating system. All station control happens via the LCU, which runs a number of control processes that communicate with the different processing boards of the station. LCU also receives station clock signals from GPS and a rubidium standard. Users can log on to the LCU via a ssh connection, and control the whole station from the command line. Single-station control is covered in more detail in Section 3, and relevant commands are listed in the command reference (Appendix A).

3 Controlling the station

3.1 Software levels

Control processes in a LOFAR station are collected in groups called software levels. The user does not need to manually start and stop a number of processes, but each level contains processes for certain actions in the station, and appropriate processes are started / stopped when entering a given level. Users can query and set the software level using the `swlevel` command. When software level is changed in

steps larger than one, the system goes through the intermediate levels as well, and starts/stops processes as needed. Processes belonging to each level are listed in the help message of the `swlevel` command. A verbatim copy of the help message is given below.

```
0:  Stop all lofar software
1:  Run Lofar daemons and PVSS
2:  Run Lowlevel hardware drivers
3:  Run Calculation services
4:  Run Hardware and software monitors
5:  Run System Health Management
6:  MAC is controlling the software
```

Software levels 0 . . . 3 are relevant in single station use, the processes belonging to the higher levels 4. . .6 are intended to be used in ILT (International LOFAR Telescope) mode.

After entering software level 2, or higher if the initial level was below 2, one needs to wait for a register update to complete. Status of the registers can be displayed with the command

`rspctl --regstat`,

which repeatedly prints several lines of text to the screen. The registers are up to date when all lines consist only of the characters '.' and '*', after which the `rspctl` process can be stopped with `ctrl-c`.

In short, beam forming (Section 3.2) requires software level 3, whereas other processes making use of RSP boards and TBBs require software level 2. All processes belonging to a software level below the current one are always available, so everything necessary for the single station use covered by this document is available if the system is running at software level 3.

3.2 Beam control

Station beamforming uses the subband data produced by the polyphase filters, and produces a number of beamformed subbands called beamlets. Beams with larger spectral width can be produced by forming several beamlets with same pointing direction but different centre frequency.[2] A LOFAR station can form up to 244/488 beamlets, each of which can have its own pointing direction and centre frequency. The user has large freedom in selecting the frequencies and pointing directions, but certain technical restrictions need to be taken into account:

1. A RCU has three inputs, only one of which can be active at a time. Inside a receiver the signal will go through an analogue band pass filter, where only one filter at a time can be selected. Due to

[2] This is only mentioned to clarify LOFAR terminology, a wideband beam still consists of separate subband signals.

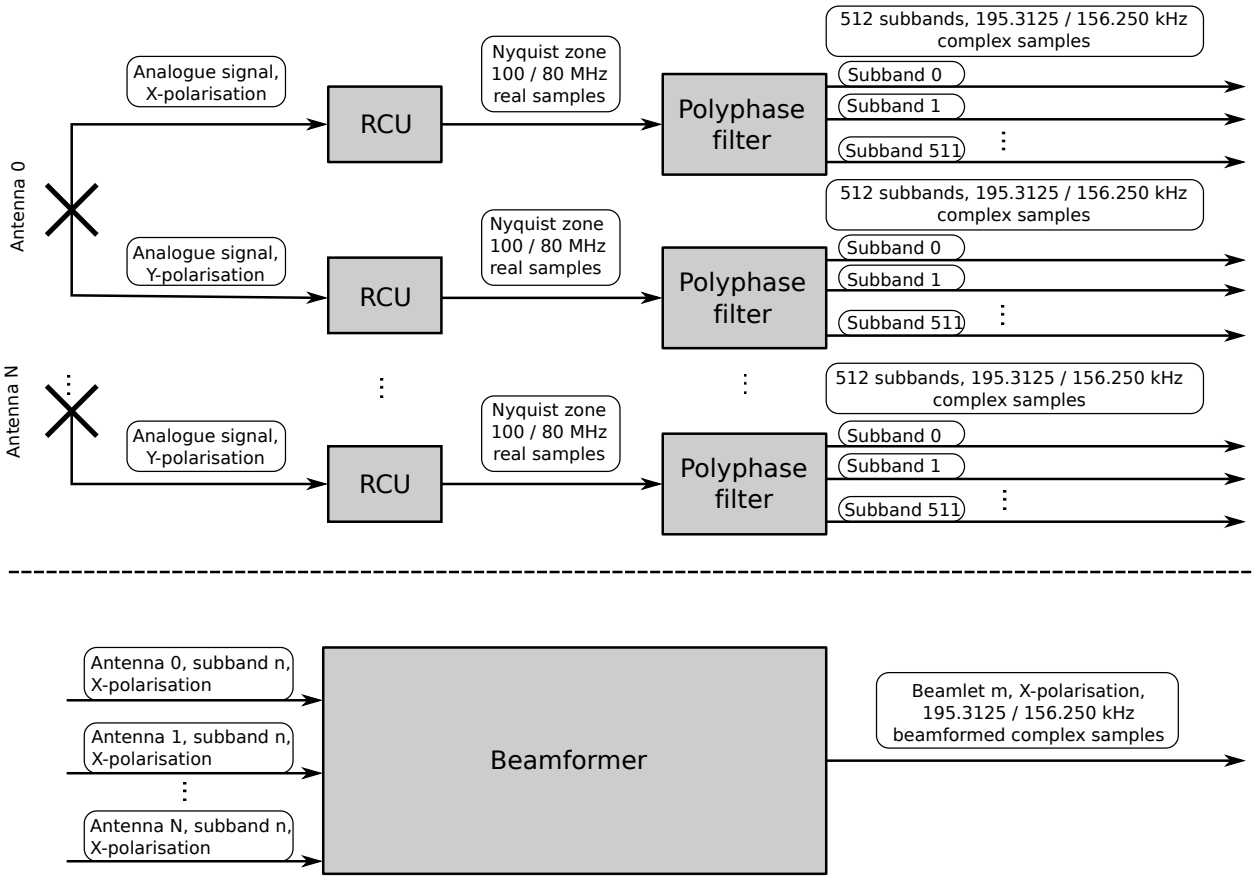

Figure 3: Functional description of signal processing steps that produce signals called subbands and beamlets. The N antennas on left are either individual LBA elements or HBA tiles. Amplified analogue signal is transferred from the antennas to RCUs, where the appropriate Nyquist zone is selected with an analogue filter, and the signal is digitised with either 200 MHz or 160 MHz sampling frequency. Signals from each antenna element are then divided into 512 narrow subbands in polyphase filters. In beamforming, the same subband from each antenna is input into a beamformer, whose output is a beamformed subband called beamlet. Both subbands and beamlets are numbered, but mapping between the subband numbers n and the beamlet numbers m is controlled by the user. A beam is a set of beamlets with a common pointing direction.

these limitations, the user must select a single receiver band (a combination of an input connector and a band pass filter) for each RCU.

2. Subbands 0 – 511 are available in principle. However, the receiver passband, i.e. passband of the analogue filter in RCUs, is always narrower than the full receiver band. A few lowest and highest subbands should thus not be used. Especially subband 0 should not be used, because it has contribution from both lowest and highest ends of the receiver band and it is vulnerable to bias voltage in the receivers.

3. Third restriction arises when using the HBA array, which makes use of the analogue beamformers inside the HBA tiles. Because the tile beam can point only to one direction at a time, all digital pointing directions must be close enough to each other, so that they fall within the tile beam. Width of a tile beam is approximately 30 degrees at 150 MHz, and it is inversely proportional to the frequency.[3] The system does not make any sanity checks about the difference between the analogue and digital beam directions, and it is on the user's responsibility to allocate the beams properly.

A single-station user has access to the station beamforming via the `beamctl` command. The command can be used when the system is running at software level 3. The `beamctl` command can handle several tasks, such as selecting receivers and receiver modes, which are accessible via the `rspctl` command as well. However, `beamctl` is the preferred way to control also the receivers, because it automatically starts and stops the receivers as needed, and automatically starts up the system slowly enough to avoid problems with high rush-in currents (see Section 3.3). The `beamctl` command is documented in more detail in Appendix A.2.

As an example, a beam with spectral width of one beamlet (subband), pointing to north at 45° elevation, at the 20th subband from the receiver band 200 – 300 MHz, could be allocated as follows:

```
beamctl --antennaset=HBA_JOINED --rcus=0:191 --rcumode=7 --subbands=20 --beamlets=0\
        --anadir=0,0.7853982,AZELGEO --digdir=0,0.7853982,AZELGEO &
```

The `--antennaset=HBA_JOINED` option selects a set of antennas called "`HBA_JOINED`", physical antennas that belong to this set are defined in the antennaset configuration file `/opt/lofar/etc/AntennaSets.conf`. The `--rcus=0:191` selects the receivers whose signal is used in beam forming. The `--rcumode=7` option selects receiver mode 7, in which HBA inputs of the RCUs are used and 200 MHz sample clock is combined with a filter passband 210 – 270 MHz. The desired subband is selected with the `--subbands=20` option. Centre frequency of subband 0 is at the lower edge of the selected receiver band, and width of a subband is 195.3125 kHz, because the 200 MHz clock is used. Centre frequency of the selected subband is thus 203.9062 MHz. The `--beamlets=0` option maps the formed beamlet to the beamlet number 0. Beamlet numbers 0 – 243 are available at all stations[4], and each beamlet must have a unique number. The last two options are pointing directions of the analogue HBA tile beamformer and the digital station beamformer, respectively. In this case the pointing directions are given as azimuth and elevation in radians. More information about available coordinate systems is given in Section 3.2.1.

The system allows a beam covering several subbands and beamlets to be allocated with a single command. As an example, a beam covering 20 subsequent subbands could be set to track a target with equatorial coordinates RA=0.4 (rad), Dec=1.2 (rad) (epoch J2000.0) using the command

[3] More information about beamwidths can be found e.g. from http://www.astron.nl/radio-observatory/astronomers/lofar-imaging-capabilities-sensitivity/lofar-imaging-capabilities/lofa and http://www.lofar.org/about-lofar/system/sensor-fields/sensor-fields

[4] When the two HBA arrays of a core station are used separately, also beamlet numbers 1000 – 1243 are used.

```
beamctl --antennaset=HBA_JOINED  --rcus=0:191 --rcumode=7 --subbands=30:49\
        --beamlets=81:100 --anadir=0.4,1.2,J2000 --digdir=0.4,1.2,J2000 &
```

The LOFAR system converts the equatorial coordinates to complex beamformer weights, and updates all digital and analogue beam pointing directions with regular intervals. The above command would thus start tracking the target automatically. The initial update after giving the `beamctl` command takes about three seconds with the HBA array, with the LBA array the delay is shorter.

New beams can be allocated to unused beamlet numbers without killing the existing beams as long as selections of antennaset, receiver mode, or analogue beam direction of HBA tiles do not cause conflicts. The `beamctl` command has some options which were not used in above examples. For a full list of options and further instructions to use them, see appendix A.2.

The system does not have a specific tool for killing individual beams allocated with the `beamctl` command. The most straightforward way to change beam directions is to kill all existing beams with

```
killall beamctl
```

and then to reallocate all of them. Naturally, an individual `beamctl` process can be killed as well, if one can identify the PID of the correct process. If the `beamctl` commands are generated via system calls from some software, there may be a simple way to get the PID as a return value. One example of such option is the python `sub.Popen` -function. A list of allocated beams together with their PIDs can also be printed with the command

```
calinfo,
```

but identifying the correct beam from the output list may not be possible.

Presently, the system does not contain any way to query the beam pointing directions. Neither is it possible to manually change the direction of an existing beam. It is thus extremely important to maintain bookkeeping of the given `beamctl` commands.

3.2.1 Coordinate systems

The present BeamServer, as well as the `beamctl` command, accept most coordinate systems that are defined in casacore (http://www.astron.nl/casacore/trunk/casacore/doc/html/index.html). Below is a list of a few available coordinate systems with examples of their use. A list of casacore coordinates can be found e.g. from (http://www.astron.nl/casacore/trunk/casacore/doc/html/classcasa_1_1MDirection.html). The examples below include only selection of the digital pointing direction, an analogue direction identical with or close enough to the digital direction needs to be selected as well, if HBA array is being used.

`J2000`	Equatorial coordinates for epoch J2000.0, right ascension and declination in radians. Track Cygnus A: `--digdir=5.233687,0.710941,J2000`
`ITRF`	Apparent topocentric position, longitude and latitude in radians. The coordinates are a position on the surface of the GRS80 ellipsoid, and the pointing direction is parallel with the position vector of the given point, i.e. parallel with a vector pointing from the centre of mass of Earth to the given point at the ellipsoid surface. Point along ellipsoid rotation axis: `--digdir=0,1.570796,ITRF`
`B1950`	Equatorial coordinates for epoch B1950.0, right ascension and declination in radians. Track Cygnus A: `--digdir=5.226131,0.7085372,B1950`
`AZELGEO`	Geodetic azimuth and elevation in radians (North through East). The angles are measured with respect to the GRS80 ellipsoid, and may be different from the azimuth and elevation measured with respect to the local horizontal plane. Point to (ellipsoid) zenith: `--digdir=0,1.5707963267948966,AZELGEO`
`GALACTIC`	Galactic longitude and latitude in radians. Track Cygnus A: `--digdir=1.329764,0.1004503,GALACTIC`
`ECLIPTIC`	Ecliptic longitude and latitude for J2000 equator and equinox. Track Cygnus A: `-digdir=5.545006,1.035425,ECLIPTIC`
`MERCURY`	Track Mercury: `--digdir=0,0,MERCURY`
`VENUS`	Track Venus: `--digdir=0,0,VENUS`
`MARS`	Track Mars: `--digdir=0,0,MARS`
`JUPITER`	Track Jupiter: `--digdir=0,0,JUPITER`
`SATURN`	Track Saturn: `--digdir=0,0,SATURN`
`URANUS`	Track Uranus: `--digdir=0,0,URANUS`
`NEPTUNE`	Track Uranus: `--digdir=0,0,NEPTUNE`
`PLUTO`	Track Pluto: `--digdir=0,0,PLUTO`
`SUN`	Track Sun: `--digdir=0,0,SUN`
`MOON`	Track Moon: `--digdir=0,0,MOON`

3.3 RSP control

Local user control of most signal processing devices in a LOFAR station, including most functionality of the RSP boards, the RCUs, and the HBA tile beamformers, is collected under the `rspctl` command.

The command allows the user to query and set several parameters, some of which are included as options in the `beamctl` command as well. When possible, it is advisable to use the `beamctl` command instead of `rspctl`, because the `beamctl` command takes automatically care of e.g. starting and stopping the relevant RCUs, and always starts up the receivers slowly enough to avoid problems with high rush-in currents. One should also notice that an allocated beam causes the system to regularly update all beamforming weights, and manual settings with `rspctl` may thus be rapidly reset by the system.[5]

If the `beamctl` command is being used to allocate beams, the user does not usually need to set any parameters with the `rspctl` command. However, `rspctl` allows the user to query several parameters, which may be interesting for checking the system status and for trouble shooting.

The `rspctl` command is first needed already before giving any `beamctl` commands for checking the register status with

`rspctl --regstat`,

as explained in Section 3.2. After allocating beams, it is useful to check that the RCUs are on and running in correct mode with the command

`rspctl --rcu`

and to check status of the RSP boards with the command

`rspctl --status`.

The command

`rspctl --subbands`

can be used to check which subbands are actually received from each RCU. The output is not completely self-explanatory; for even-numbered RCUs the subband numbers are multiplied by two, and for odd-numbered RCUs the subband numbers are multiplied by two and the result is incremented by 1. As an example, subband 10 is denoted as 20 in output of even-numbered RCUs and as 21 in output of odd-numbered RCUs. `rspctl --subbands` also prints the total number of subbands produced by each RCU, as value of the variable "`subbands`".

Spectra from individual RCUs (antenna elements) can be monitored with the command

`rspctl --statistics`,

which repeatedly plots power spectra of all receivers.[6] It is also possible to plot the signal powers of each

[5] If this would be a problem in some special experiments, the update interval can be increased by changing the values in `/opt/lofar/etc/BeamServer.conf`.

[6] The user must have the X Window System installed on her/his computer, and X11 forwarding needs to be enabled in

beamlet by using the command

```
rspctl --statistics=beamlet
```

or array covariance matrix at a selected subband with

```
rspctl --xcsubband=<int>
rspctl --xcstatistics.
```

The above commands also allow the signal statistics to be recorded in files. Data recording is covered in more detail in Sections 4.1, 4.2, and 4.3. Users who want to send raw beamlet data to CEP or some other network device can switch the UDP data stream off/on and query its status with

```
rspctl --datastream=[0|1|2|3].
```

Only the options `0` (off) and `1` (on) are used in international stations. The options `2` and `3` are used in core stations, where the ring of RSP boards can be split in two halves, and data output from both of these need to be controlled separately. See Appendix A.3 for more details.

If the user wants to use the receivers without forming beams, they need to be started manually with a couple of `rspctl` commands. In order to guarantee that the beam server does not reset the user-defined settings, one should begin with switching to software level 2, which disables the beam server,

```
swlevel 2,
```

and then wait for register update, i.e. wait until the output of

```
rspctl --regstat
```

consists only of the characters '*' and '.'.

The current receiver mode can be queried with

```
rspctl --rcu.
```

If one wants to switch to one of the receiver modes 0 – 7 (receiver modes are listed in Appendix A.2), the mode can be changed with

```
rspctl --rcumode=[0|1|2|3|4|5|6|7].
```

After the previous commands the receivers are running in correct mode, but the data stream from the receivers to the RSP boards still needs to be started with

the ssh-connection to the LCU. Use `ssh -X` in unix-alikes.

©ASTRON 2018

`rspctl --rcuenable=[1|0]`.

If running in receiver mode 5, the received spectrum will be inverted, and one might be interested in inverting it back with

`rspctl --specinv=[1|0]`.

Notice that this command is not needed if the receiver mode is selected with `beamctl`, which automatically inverts the spectrum if receiver mode 5 is selected.

Remark: The above commands are replaced with a new combined command.

`rspctl --mode=[0|1|2|3|4|5|6|7]`.

This command checks and sets automatically the correct rcumode, rcuenable, spectal inversion (when needed) and switches sample clock 160/200 MHz when needed. Also problems with high inrush currents in the HBA during power up phase are solved (also for rcumode) by enabeling only one HBA tile every 5 mS.

A new command is also available to switch between 4,8 and 16 bit beamforming mode. Due to beampower clipping in 4 bit mode we recommend to only use the 8 and 16 bit mode.

Remark: The bitmode command needs to be set before the beamctl command is used!

`rspctl --bitmode=[4|8|16]`.

After these commands one can check that the system is running properly with the commands

`rspctl --rcu`

`rspctl --status`

`rspctl --statistics`.

The `rspctl` command allows the user to set and query numerous parameters, see Appendix A.3 for a complete listing.

3.4 TBB control

Local user control of the transient buffer boards is collected under the `tbbctl` command. It allows one to start and stop writing to the ring buffers, to send data to CEP or write it to files, and to set triggering systems that monitor incoming data. Because the TBBs are designed solely for data recording purposes,

their usage is covered together with other data recording methods in Section 4.4.

4 Data recording and transmission

This section covers methods for recording different kinds of data products with a single LOFAR station. In addition, instructions for controlling the data output to CEP are given. The station itself offers the possibility to continuously record so-called subband statistics (SST), beamlet statistics (BST), and array covariance matrices (ACM).

The term subband statistics refers to average power in a subband produced by the polyphase filter. Statistics of all 512 subbands can be recorded from all RCUs of a station simultaneously. Beamlet statistics refers to similar statistics of beamformed data, which can be recorded from all 244/488 beamlets produced by a station simultaneously. The array covariances, also known as "crosslets" or "visibilities", are covariances of signals from different RCUs. ACMs can be calculated and stored for one selected subband at a time. All these data products can be recorded with one second or coarser time-resolution.

In addition to the above mentioned correlated data products, the TBBs can be used for recording short periods of raw voltage data. This data can be either raw real samples produced by the A/D converter, or complex samples of subband data produced by the polyphase filter. The TBBs are not designed for recording long continuous time series, in their normal use the time-series read from the buffer are less than a millisecond long [6].

In order to record continuos stream of voltage level data from a LOFAR station, one needs to capture the packets that the station transmits to central processor. This task is not part of normal LOFAR operations, which is the reason for lack of standard software tools for the CEP data recording. A small example about recording the CEP output data with Wireshark is included in this document as an appendix (C). Together with the data output format descriptions in sections (4.5) and (4.6), it should provide the necessary information for users interested in building more sophisticated data acquisition systems.

4.1 Recording subband statistics

Subband statistics refers to average power in each of the subbands produced by the polyphase filter, i.e. to signal spectrum over the whole receiver band. The `rspctl` command can be used either for repeatedly plotting the subband statistics, or alternatively for writing the subband statistics to files in LCU hard drive. The statistics can be plotted with the command

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

where `subband` is the default value for `--statistics` and could be omitted in this case. The optional

`--integration` option is time-resolution in seconds, the default is one second. The `--select` option allows one to select the RCUs whose output is plotted, the default is to select all RCUs of a station.

The same data that can be plotted with the above command can be alternatively written to files by defining a duration for the recording

```
rspctl --statistics --duration=<seconds>
      [--integration=<seconds> --select=<set> --directory=<directory>]
```

The optional `--directory` option directs the output to the given directory, the default is to use the current working directory. It is advisable to use the directory `/localhome/data/`, because all users should have write privileges to this directory.

As an example, one hour of subband statistics with five second time-resolution could be recorded by the command

```
nohup rspctl --statistics --duration=3600 --integration=5\
             --directory=/localhome/data/sst_test > rspctl.log 2>&1 &
```

where the `nohup` command has been added to guarantee that the `rspctl` process will continue running if connection to the LCU is lost, and all output messages are redirected to the file `rspctl.log`. The last `&` puts the process to background.

The file format of subband statistics files is explained in Section 5.1.

4.2 Recording beamlet statistics

Beamlet statistics refers to average power in each beamlet. The `rspctl` command can be used for recording the beamlet statistics in the same way as it can be used for recording the subband statistics (Section 4.1), if beamlet statistics is selected with `--statistics=beamlet` option. One hour long recording of statistics of all beamlets, with five second time-resolution, could be recorded with the command

```
nohup rspctl --statistics=beamlet --duration=3600 --integration=5\
             --sel=0:3 --directory=/localhome/data/bst_test > rspctl.log 2>&1 &
```

Notice that the `--select` option selects an output SERDES ring [0,1] and polarisation in this case. Only Dutch core stations have two data rings by means of a data splitter. For ILT and remote stations `--select=0,1` is sufficient for both X- and Y- polarisation. When no selection is done four data files will be written with the last two files containing zeros for ILT or remote station.

The beamlet statistics plot from all 244/488 beamlets of a station are the amplitude/phase corrected sum of all antennas defined in the beam. The statistics plot contains one line for the X- and one for the Y-beam power. For core stations with splitter enabled you get two seperate beamlet statistics plots. One plot for ring zero and one for ring one. **Remark: Only in beamformed mode (BST) the station calibration table is applied to the antenna signals!**. A mapping of the lanes and beamlets in 8- and 16 bits is given in Table 3.

The file format of subband statistics files is explained in Section 5.2.

4.3 Recording array covariances

Array covariances (crosslets, visibilities) can be calculated only for one subband at a time. The subband can be selected with

`rspctl --xcsubband=<int>`.

The array covariances can be plotted and recorded with a command very similar to that used for subband and beamlet statistics.

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

Because station calibration requires a large amount of array covariances from all subbands, a LOFAR station can be configured to continuously loop over all subbands, and to record the covariances from each subband with one second time-resolution. A data set containing an array covariance matrix from each subband is called an "array covariance cube" (ACC). Instructions for calibration data recording are given in Station Calibration Cookbook [18].

The file format of array covariance files is explained in Section 5.3.

4.4 Recording data with TBBs

Transient buffer boards can be used for recording raw samples from the A/D converter or subband data produced by the polyphase filter. Both of these data products are actually transferred to the TBBs via RSP boards, and selection between raw A/D converter data and subband data is thus under the `rspctl` command. Raw A/D converter samples are selected with

`rspctl --tbbmode=transient`

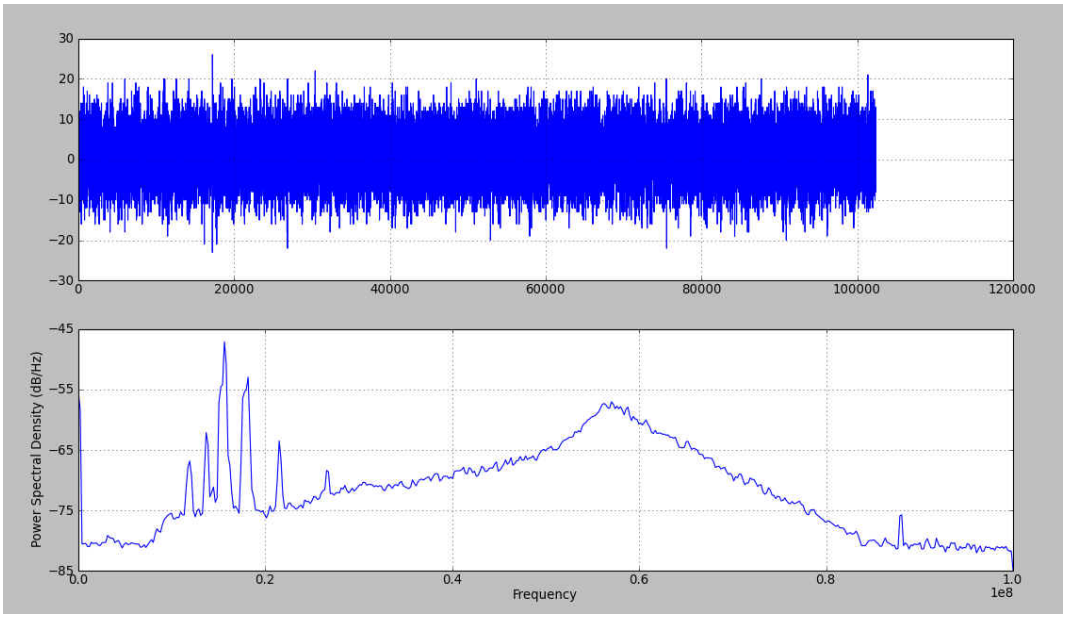

Figure 4: A time series of raw A/D converter data from LBA (upper panel), and its power spectrum (lower panel) ©Menno Norden.

and subband data with

`rspctl --tbbmode=subbands,<set>`,

where `<set>` is the set of subbands to be recorded, e.g. `rspctl --tbbmode=subbands,0:9`. Notice that `rspctl` allows one to select any combination of subbands, but the data frames used for transferring data from RSP boards to TBBs can handle maximum 487 subbands [14]. Selecting a larger number of subbands may lead to errors in TBB output.

The above command selects the data type to be recorded, but does not start the actual buffering. Controlling the buffering and reading the data from the buffer happens with the `tbbctl` command.

Before starting the recording, one needs to allocate memory buffers with

`tbbctl --alloc [--select=<rcuset>]`,

where the optional `--select` allows one to allocate memory for the selected set of RCUs only. All available memory will be divided between the selected RCUs, and selecting a smaller number of RCUs will thus enlarge the allocated buffer size. When called without the `--select` option, `tbbctl --alloc` will allocate memory for all RCUs of a station.

The command

Table 2: Trigger log file contents for different versions of `tbbctl`.

Column	version $\leq$ 2.39	version $\geq$ 2.40
1	RCU	RCU
2	Sequence number	Full second
3	Full second	Nanosecond
4	Sample number	Trigger sum
5	Trigger sum	Trigger samples
6	Trigger samples	Peak value
7	Peak value	Power before
8	Power before	Power after
9	Power after	Missed
10	Missed	

`tbbctl --rcuinfo`

allows the user to query the status of all RCUs, for which memory has been allocated.

After memory allocation one can start recording with

`tbbctl --record [--select=<rcuset>]`,

where the optional `--select` allows one to start recording from a subset of all RCUs. When omitted, the default is to record from all RCUs for which memory buffer has been allocated. When the recording has been started, the system will write data to a ring buffer until the recording is stopped by

`tbbctl --stop [--select=<rcuset>]`.

LOFAR has also a special triggering system, which signals the LCU of a station when a triggering event has been detected. Notice that by default the LCU will not stop recording, but it will only store contents of the trigger message to a file in `/locahome/data`. One file per day is normally produced, whose name consists of a date string and the word "TRIGGER". In order to actually stop recording after receiving a trigger signal, one needs a system that listens for the trigger signals and sends stop signals to TBBs as necessary. Such a system could be e.g. a simple script monitoring the contents of the trigger log file.

Depending on `tbbctl` version, the trigger file has either ten or nine columns. Contents of the columns are listed in (Table 2). The version number can be queried with `tbbctl --version`.

Trigger signals can be generated by CEP, by a triggering system running in the TBBs, or by an external triggering system connected to the TBBs. It is also possible for a user to manually generate trigger

signals from LCU command line.

The local triggering system inspects the data arriving to TBBs, and signals a trigger if a strong enough transient pulse is detected. Local control of the trigger system is collected under options of the `tbbctl` command. The triggering system is initialised with the commands

```
tbbctl --setup=level,start,stop,filter,window,mode
```

and

```
tbbctl --coef=f00,f01,f02,f03,f10,f11,f12,f13,
```

which are documented in more detail in Appendix A. One can then start recording with

```
tbbctl --record
```

and release the triggering system, i.e. start inspecting the incoming data, with

```
tbbctl --release.
```

Arriving triggering messages can be monitored from LCU with the command

```
tbbctl --listen=[one_shot | continues],
```

which prints contents of received trigger signals to screen.

Information of the latest trigger event of a RCU can be queried with

```
tbbctl --triginfo=rcu.
```

Two options are available for reading data from TBB memory: the data can either be sent to central processing (Section 4.5), or it can be written locally to LCU hard drive.

Data are copied to local files by first stopping the recording with

```
tbbctl --stop,
```

and then giving the command

```
tbbctl --readpage=rcunr,startpage,npages,
```

where `rcunr` identifies the RCU whose data one wants to read, and `npages` memory pages will be read, starting from `startpage`. Notice that the file output has been designed for test purposes only, and it is very slow.

After recording data, one can free the allocated memory with the command

`tbbctl --free [--select=<rcuset>].`

The format of TBB data files recorded to LCU hard drive is explained in Section 5.4.

4.5 TBB data CEP output

Two commands are available for CEP data transmission, the

`tbbctl --readall=pages [--select=rcus]`

command will read the given amount of latest memory pages from selected RCUs. It can be given any time when data are being recorded. The command will stop the recording, read the data, and restart the recording after reading the memory.

Another option is the

`tbbctl --read=rcunr,secondstime,sampletime,prepages,postpages`

which reads data from the selected RCU. The input contains time in seconds, number of samples after start of the second, and page offsets before and after the given time.[7] Unlike `--readall`, the `--read` option requires that the recording is manually stopped with

`tbbctl --stop`

before reading.

The format of data packets transmitted to CEP is explained in Section 5.5.

4.6 Beamlet data CEP output

When beam server is running, i.e. when running the system at software level 3 or higher, beamlet CEP data packets are automatically produced by the four RSP boards[8] that are endpoints of the beamforming lanes. A user can query status of the data stream or switch it on/off with the command

`rspctl --datastream=[0|1|2|3].`

[7] The option works this way in `tbbctl` versions 2.39 and lower, in higher versions all times and offsets are given in seconds and nanoseconds.

[8] Eight in core stations when the HBA0 and HBA1 are used separately.

The data stream is off by default, and the user needs to switch it on manually using the above command.

5 Data output formats

Data are output from RSP boards and TBBs in two different ways: some data are being transmitted to central processing, whereas options for writing data to LCU hard drive are available as well. Raw voltage level beam formed data are the only data product that cannot be written directly to files, but the station can only output it via a network connection. For TBB data there are options for both local data recording and CEP transmission, but the file recording option is only for testing purposes, and not fast enough for recording significant amounts of TBB data. Subband statistics, beamlet statistics, and array covariances (visibilities) can be recorded only to LCU hard drive. Exact definitions of the different data output formats are given in this Section.

5.1 Subband statistics file format

Data from each individual RCU is written to a separate file. The file name contains a timestamp, the string "sst" to indicate that the file contains subband statistics data, and the RCU number. As an example, the file

```
20111205_105031_sst_rcu000.dat
```

was recorded December 5 2011 10:50:31 UT from RCU 0. The data are written as 64-bit little-endian floats without any header. One data point per integration period is produced.

If the recording is started with the `--select=<set>` option, only the selected RCUs will be recorded.

Notice that, with the present software version, the data output from RSP boards starts slowly, and four seconds of data are missing from all SST recordings. It is unclear that which exact samples are missing, and there is thus a four second ambiguity in timestamps of individual data samples. If duration of the recording is shorter than five seconds and one second time-resolution is used, one data sample will be produced, but it is unclear that which period of time it represents.

5.2 Beamlet statistics file format

Each data ring [0—1] produces two files, one for X-polarisation and another for Y-polarisation. The data are written as 64-bit little-endian floats without any header. The file name contains a timestamp, the string "bst" to indicate that the file contains beamlet statistics data, number of the ring, and 'X' or 'Y' indicating the polarisation. As an example, the file

Table 3: Mapping between lane numbers and beamlet numbers. All four lanes will output the BST data of all beamlets, but only the beamlets listed in the table are final beamforming samples.

Lane	Beamlets (16 bit)	Beamlets (8 bit)
0	0 – 60	0 –121
1	61 – 121	122 – 243
2	122 – 182	244 – 365
3	183 – 243	366 – 487

`20111205_105947_bst_00Y.dat`

was recorded December 5 2011 10:59:47 UT from ring 0, and it contains Y-polarisation data.

If the recording is started with the `--select=<set>` option, data are recorded only from the selected ring and polarisation.

Each of the selected rings will output 244/488 data samples per integration period. They are written to the files one integration period at a time, always starting from the beamlet number 0.

5.3 Array covariance file format

Array covariance data are written in a single file. The data are written as 64-bit little-endian floats without any header. The file name contains a timestamp and the string "xst" to indicate that it contains crosslet statistics (covariance, visibility) data. As an example, the file

`20111205_114317_xst.dat`

was recorded December 5 2011 11:43:17 UT.

If the station has N_{rcu} RCUs, an $N_{rcu} \times N_{rcu}$ complex array covariance matrix will be produced from each integration period. Denoting the covariance of receivers i and j with $x_{i,j}$, the data points of single integration period are written to the file in the order

$$Re(x_{0,0}), Im(x_{0,0}), Re(x_{0,1}), Im(x_{0,1}), Re(x_{0,2}), Im(x_{0,2}), \ldots, Re(x_{N_{rcu},N_{rcu}}), Im(x_{N_{rcu},N_{rcu}}),$$

where Re and Im denote the real and imaginary parts of the complex data values.

If array covariances are recorded for station calibration, as explained in [18], the files will be different. In the file name, the timestamp will be followed with "acc" ("array covariance cube"), and dimensions of the covariance matrices stored in the file. As an example, the file

`20111209_104029_acc_512x192x192.dat`

was recorded December 9 2011 10:40:29 UT, and it contains 192 × 192 covariance matrices from 512 subbands. 192 is the number of LBA elements in the station (LBA was used in the experiment). Covariance matrices from each subband are recorded subsequently with increasing subband number, i.e. first 192 x 192 = 36864 complex samples are from subband 0, the next 36864 samples from subband 1, etc.

5.4 TBB data file format

The `tbbctl --readpage` command copies contents of TBB memory pages to files. In TBB memory each page consists of a header and a payload, whose contents are documented in detail in [14]. Information in the TBB memory pages is copied to the files as such, but transient data that is stored as 12-bit integers in the TBB memory is extended to 16-bit format. The header is always 88 octets long, its contents are listed in Table 4.

Table 4: TBB page header layout. The values in octets 0 – 19 are unsigned little-endian integers.

octets	contents
0	Station ID
1	RSP ID
2	RCU ID
3	Sample clock frequency
4 – 7	Sequence number
8 – 11	Time, full second of first sample
12 – 15	Sample / slice number in current 1s interval
16 – 17	Number of samples in the frame payload
18 – 19	Number of frequency bands
20 – 83	Band selection. Each bit tells whether the subband with the bit index is present or not.
84 – 85	Reserved for future use, set to 0
86 – 87	Header checksum

The page header is immediately followed by the payload, whose length is given in octets 16 – 17 of the header. The payload is immediately followed by a four octets long checksum. The data are written in the order header 0, payload 0, checksum 0, header 1, payload 1, checksum 1, header 2, ... The payload data are written as signed 16-bit little-endian integers.

If only a small number of subbands is recorded with TBB, several time-slices of data may be stored in a single memory page. In this case the slices are written subsequently, i.e. all subbands of slice 0 are written before slice 1, etc.

5.5 TBB CEP data packet format

Table 5: TBB data packet layout. Octet count zero is set to the first octets of the ethernet header, the destination MAC address. TBB page payload (octets 130 – 2177) format is given assuming that transient data are being recorded. In case of subband data, the real and imaginary parts of a sample are written subsequently with the real part written first, both as 16-bit integers. If only a few subbands are recorded, the same TBB page may contain several time-slices, which are written subsequently also in the CEP data packets.

octets	content
-8 – -2	Ethernet frame preamble
-1	Start of frame delimiter
0 – 5	MAC destination
8 – 11	MAC source
12 – 13	Ethernet type
14	Version & header length
15	Type of service
16 – 27	Total length
18 – 19	Identification
20 – 21	Flags (2 bits) and fragment offset
22	Time to live
23	Protocol
24 – 25	Header checksum
26 – 29	Source IP
30 – 33	Destination IP
34 – 35	UDP source port
36 – 37	UDP destination port
38 – 39	UDP length
40 – 41	UDP checksum
42	Station ID

Octets	Contents
43	RSP info
44	RCU ID
45	Sample clock frequency
46 – 49	Sequence number
50 – 53	Time, full second of first sample
54 – 57	Sample / slice number in current 1s interval
58 – 59	Number of samples in the frame payload
60 – 61	Number of frequency bands (0 if transient data)
62 – 125	Band selection. Each bit tells whether the subband with the bit index is present or not
126 – 127	Reserved for future use, set to 0
128 – 129	TBB page header checksum
130 – 131	Sample 0
132 – 133	Sample 1
⋮	⋮
2176 – 2177	Sample 1024
2178 – 2181	Ethernet frame checksum

TBB data transfer to CEP uses the UDP/IP protocol on top of the Ethernet protocol [14]. Likewise when writing TBB data to files, the TBB memory page headers are copied as such, and the data that was possibly stored in 12-bit format in the TBB memory is extended to 16-bit format.

5.6 Beamlet CEP data packet format

Beamlet data are transmitted from the station to CEP using the UDP/IP protocol on top of the Ethernet protocol [11]. The data transmission makes use of jumbo ethernet frames, which allow payload size up

Table 6: CEP beamlet data packet layout. Octet count zero is set to the first octets of the ethernet header, the destination MAC address. Length of the payload (octets 58 – 7865) is calculated assuming that 16 time-slices of 61 beamlets are included in the packet. If smaller amount of beamlets or slices is included, the payload will be shortened correspondingly. Payload length can be calculated e.g. from the number of beamlets and number of blocks (slices), which are stored in octets 48 – 53.

octets	content
-8 – -2	Ethernet frame preamble
-1	Start of frame delimiter
0 – 5	MAC destination
8 – 11	MAC source
12 – 13	Ethernet type
14	Version & header length
15	Type of service
16 – 27	Total length
18 – 19	Identification
20 – 21	Flags (2 bits) and fragment offset
22	Time to live
23	Protocol
24 – 25	Header checksum
26 – 29	Source IP
30 – 33	Destination IP
34 – 35	UDP source port
36 – 37	UDP destination port
38 – 39	UDP length
40 – 41	UDP checksum
42	Version ID
43	Source info
44 – 45	Configuration
46 – 47	Station ID
48	Number of beamlets
49	Number of blocks
50 – 53	Timestamp
54 – 57	Block sequence number

Octets	Contents
58 – 59	Beamlet 0, slice 0, x-polarisation, real part
60 – 61	Beamlet 0, slice 0, x-polarisation, imaginary part
62 – 63	Beamlet 0, slice 0, y-polarisation, real part
64 – 65	Beamlet 0, slice 0, y-polarisation, imaginary part
66 – 67	Beamlet 0, slice 1, x-polarisation, real part
⋮	⋮
184 – 185	beamlet 0, slice 15, y-polarisation, imaginary part
186 – 187	beamlet 1, slice 0, x-polarisation, real part
⋮	⋮
7864 – 7865	beamlet 60, slice 15, y-polarisation, imaginary part
7866 – 7869	Ethernet frame checksum

to 9000 bytes. A detailed description of the data packet format is given in [11]. Starting from the beginning, each packet contains a 22-byte ethernet header, followed by a 28-byte transport header, a 16-byte application header, the actual beamlet data, and finally a 4-byte checksum of the ethernet frame. Contents of a CEP beamlet data package are listed in Table 6.

The ethernet frames are formed in and transmitted from the four RSP boards that are end points of the beamforming lanes. Because each lane produces 61 beamlets, each of the output RSP boards transmits 61 beamlets. Assuming that lanes 0 – 3 end on RSP boards 0 – 3, RSP board 0 transmits beamlets 0 – 60, board one beamlets 61 – 121, board 2 beamlets 122 – 182, and board 3 beamlets 183 – 243. In core stations there is a possibility to form 488 beamlets when HBA0 and HBA1 arrays are used separately. Four additional output RSP boards will then be defined in `/opt/lofar/etc/RSPDriver.conf`. These boards will output beamlet numbers 1000 – 1060, 1061 – 1121, 1122 – 1182, and 1183 – 1244.

The data packets do not contain information of the beamlet numbers, subband numbers, or pointing directions. The beamlets in a packet can be identified based on the source address – which identifies the RSP board – and knowledge of the `beamctl` commands that have been given. Source and destination addresses of the data packets are defined in the configuration file `/opt/lofar/etc/RSPDriver.conf`.

It is possible to limit the number of beamlets transmitted from each RSP board by modifying the entry `RSPDDriver.CDO_N_BEAMLETS` in file `/opt/lofar/etc/RSPDriver.conf`. If the number of beamlets is smaller than 61, each RSP board will transmit the given number of beamlets, starting from its lowest beamlet number. As an example, if one sets `RSPDDriver.CDO_N_BEAMLETS=10`, board 0 will transmit beamlets 0 – 9, board 1 beamlets 61 – 70, board 2 beamlets 122 – 131, and board 3 beamlets 183 – 192. Other beamlets will not be transmitted at all, and their data will be lost.

Each UDP packet contains equal number of time slices, by default 16 slices. One time slice of data contains one sample from each beamlet. The number of time slices per packet is defined in the entry `RSPDriver.CDO_N_BLOCKS` in the file `/opt/lofar/etc/RSPDriver.conf`.

The actual beamlet data are written to the payload as signed little-endian 16-bit integers. The samples are written in beamlet order, i.e. all samples from a beamlet are written subsequently, starting from the lowest numbered beamlet of the lane. A time-slice of a beamlet contains two complex samples, one from x- and another from y-polarisation. As an example, the payload listing in Table 6 is the default output from RSP board 0.

References

[1] M. Arts. Effects of holes in bow-tie antenna. Technical Report LOFAR-ASTRON-RPT-119, ASTRON, Dwingeloo (The Netherlands), December 2004.

[2] M. J. Arts. Overview of antenna types for hba. Technical Report LOFAR-ASTRON-RPT-118, ASTRON, Dwingeloo (The Netherlands), October 2004.

[3] M. A. Brentjens. LOFAR reference plane and reference direction. Technical Report LOFAR-ASTRON-MEM-248, ASTRON, Dwingeloo (The Netherlands), December 2008.

[4] A. Doorduin. TBB Transient Detector User Manual. Technical Report LOFAR-ASTRON-DDD-056, ASTRON, Dwingeloo (The Netherlands), November 2009.

[5] F. J. Harris, C. Dick, and M. Rice. Digital Receivers and Transmitters Using Polyphase Filter Banks for Wireless Communications. *IEEE Transactions on Microwave Theory and Techniques*, 51 (4):1395–1412, 2003. doi: 10.1109/TMTT.2003.809176.

[6] A. Horneffer and S. Lafèbre. The LOFAR VHECR Trigger. Technical Report LOFAR-ASTRON-SDD-055, ASTRON, Dwingeloo (The Netherlands), March 2007.

[7] G. W. Kant, M. J. Norden, and E. van der Wal. HBA Requirements Specification. Technical Report LOFAR-ASTRON-SRS-018, ASTRON, Dwingeloo (The Netherlands), April 2007.

[8] H. Kollen and M. Bentum. LOFAR Glossary of Terms and Abberviations. Technical Report LOFAR-ASTRON-RPT-002, ASTRON, Dwingeloo (The Netherlands), March 2007.

[9] E. Kooistra. Brief description of the LOFAR station signal processing. Technical Report LOFAR-ASTRON-MEM-238, ASTRON, Dwingeloo (The Netherlands), October 2007.

[10] W. Lubberhuizen and E. Kooistra. RSP Firmware Functional Specification. Technical Report LOFAR-ASTRON-SDD-001, ASTRON, Dwingeloo (The Netherlands), March 2007.

[11] W. Lubberhuizen and E. Kooistra. RSP - CEP Beamlet Data Interface. Technical Report LOFAR-ASTRON-SDD-009, ASTRON, Dwingeloo (The Netherlands), September 2009.

[12] M. J. Norden. LOFAR Station User Manual. Technical Report LOFAR-ASTRON-MEM-239, ASTRON, Dwingeloo (The Netherlands), April 2009.

[13] M. J. Norden and G. W. Kant. RCU II Signal Analysis and Specification. Technical Report LOFAR-ASTRON-SRS-014, ASTRON, Dwingeloo (The Netherlands), March 2007.

[14] W. Poiesz and A. Doorduin. TBB Design Description. Technical Report LOFAR-ASTRON-SDD-047, ASTRON, Dwingeloo (The Netherlands), November 2009.

[15] W. Poiesz, E. Kooistra, and W. Lubberhuizen. RSP Firmware Design Description. Technical Report LOFAR-ASTRON-SDD-018, ASTRON, Dwingeloo (The Netherlands), September 2009.

[16] W. A. van Cappellen, M. Ruiter, and G. W. Kant. Low Band Antenna Architectural Design Document. Technical Report LOFAR-ASTRON-ADD-009, ASTRON, Dwingeloo (The Netherlands), March 2007.

[17] S. J. Wijnholds. LBA station configuration. Technical Report LOFAR-ASTRON-MEM-241, ASTRON, Dwingeloo (The Netherlands), December 2007.

[18] S. J. Wijnholds. Station Calibration Cookbook. Technical Report LOFAR-ASTRON-MAN-049, ASTRON, Dwingeloo (The Netherlands), October 2011.

©ASTRON 2018

A LOFAR station command reference

This appendix contains a list of commands relevant for a single-station user. Some of the command-line arguments are explained in more detail than in the standard help messages of the functions. Together with Appendix B it should form a rather comprehensive guide for locally controlling a LOFAR station.

A.1 swlevel

The `swlevel` command is mainly used to query and set the software level of a station. In this usual form the command takes only one optional argument, the new software level. When called without an argument, swlevel prints a list of available levels and processes that belong to them, together with the PID of a running process, or the keyword DOWN for processes that are not running.

Below is the formal syntax of the `swlevel` command, copied from the help message (`swlevel -h`).

```
swlevel [-s/S/v/V/u/U] [-i x] [-r/q processname] [-l][ 0 |1 |2 |3 |4 |5 |6 ]
```

`-s`	Show current level, exit (-S: only print number)
`-v`	Show running LOFAR version, exit (-V: only print version)
`-u`	Show users owning running processes (-U: same)
`-i`	Load RSP firmware from image [x] (default image is 1)
`-l`	Set to level as provided (optional)
`-q`	Quit process with name processname
`-r`	Run process with name processname

A.2 beamctl

All functionality regarding beamforming can be controlled with the `beamctl` command. Beams are allocated for the LBA array with

```
beamctl <rcuspec> <dataspec> <digpointing>
```

and for the HBA array with

```
beamctl <rcuspec> <dataspec> <anapointing> <digpointing>
```

info on the calibration tables

```
beamctl --calinfo
```

where:

```
<rcuspec>      = --antennaset --rcus --band
<dataspec>     = --subbands --beamlets
<digpointing>  = --digdir
<anapointing>  = --anadir
```

The options are used as follows:

`--antennaset=name` Antenna set selection. Accepts any antenna set name that is defined in the file `/opt/lofar/etc/AntennaSets.conf`.
Examples: `--antennaset=HBA_JOINED --antennaset=LBA_INNER`
Examples: `--antennaset=LBA_SPARSE_ODD --antennaset=LBA_SPARSE_EVEN`
NOTICE: `--antennaset` must match with the `--band` option. A mismatch will not be detected by `beamctl` and will cause useless output data.
NOTICE: When using `--antennaset=HBA_DUAL` at the core stations, one needs to manually switch the serdes splitter on (see `rspctl --splitter`) before giving the `beamctl` command. Beamlets produced from the HBA1 array are then mapped to beamlet numbers 1000 – 1243.

`--rcus=<set>` Receiver selection. Accepts any integer values between 0 and number of available RCUs. Number of RCUs is 96 in remote and core stations, and 192 in international stations.
Examples: `--rcus=0:191 --rcus=0:41,60,61,80:95`
NOTICE: When using the serdes splitter in core stations, only RCUs 0 – 47 are available with HBA0 and RCUs 48 – 95 with HBA1.

`--band=name` The RCU filter selection and mode. For LBA name = `10_90 or 30_90`. For HBA name = `110_190, 170_230 or 210_250`. The name of band selection, may not conflict with antennaset.

`--subbands=<set>` Subband selection. Accepts any integers between 0 and 511.
Examples: `--subbands=100:200`, `--subbands=100:150,155:200,210`
NOTICE: Subband 0 should not be used, because it gets contribution from both the lowest and highest frequencies of the selected frequency band.
NOTICE: The 100 MHz / 80 MHz bandwidth of the digital receiver is wider than the passband of the analogue filters in RCUs. Make sure that the selected subbands fall inside the passband of the analogue filter (see the table for `--rcumode`).

Table 7: LOFAR station receiver modes. Nominal frequency range is the RCU frequency coverage given in LOFAR documentation [e.g. 12]. Actual passband is the frequency range which falls inside the receiver filter passband [13]. The values in the table are measured -3dB points, except in modes 1 and 3, where more attenuation is allowed in between 10 and 30 MHz [13]. In case of modes 5 and 7, notice that the HBA array frequency range is defined to be only 120 - 240 MHz [7]. Receiver band is the full frequency coverage of the selected Nyquist zone, i.e. the range of frequencies covered by the 512 subband signals.

Mode	RCU input	Nominal frequency range [MHz]	Approximate actual passpand [MHz]	Receiver band [MHz]
0	–	–	–	–
1	LBL	10 – 90	10 – 80	0 – 100
2	LBL	30 – 90	30 – 80	0 – 100
3	LBH	10 – 90	10 – 80	0 – 100
4	LBH	30 – 90	30 – 80	0 – 100
5	HBA	110 – 190	115 – 185	200 – 100 (inverted)
6	HBA	170 – 230	170 – 220	160 – 240
7	HBA	210 – 270	215 – 260	200 – 300

NOTICE: The number of subbands must be equal to the number of beamlets (see `--beamlets`)

`--beamlets=<set>` Beamlet number selection. Accepts any integer from 0 to 243 [0..487 in 8 bit mode] at all stations. The subbands selected with the `--subbands` option are mapped to the given beamlet numbers. Beamlet numbers 0 – 60 [0:121] belong to lane 0 and are usually output from RSP board 0, beamlet numbers 61 – 121 [122:243] belong to lane 1 and are output from RSP board 1, etc. At core stations beamlet numbers 1000 – 1243 are used for beamlets from the HBA1 array, when the serdes splitter is switched on. Their data are output from different RSP boards than the "normal" beamlets 0 – 243, in standard configuration from RSP boards 6, 7, 8, and 9.
Examples: `--beamlets=0:100`, `--beamlets=10:60,65:101`
NOTICE: The number of beamlets must be equal to the number of subbands (see `--subbands`).

`--digdir=longitude,latitude,type[,duration]` Pointing direction of the digital station beam, in any of the coordinate systems understood by LOFAR (see Section 3.2.1). The pointing direction is updated with regular intervals, whose length is defined in the configuration file `/opt/lofar/etc/BeamServer.conf`. The values `longitude,latitude` are either longitudes and latitudes in their usual meaning, or azimuths and elevations, depending on the coordinate system. `type` is the name of the coordinate system. The optional `duration` is lifetime of the beam in seconds.
Examples: `--digdir=0,1.570796,AZELGEO` `--digdir=5.2336865,0.71094095,J2000`

NOTICE: When using the HBA array, make sure that the digital beam direction falls inside the analogue tile beam, whose direction given with the `--anadir` option. Tile beam width is approximately 30 degrees at 150 MHz, and it is inversely proportional to frequency.
NOTICE: If `duration` is given, the data flow will stop after the given time, but the `beamctl` process will continue running, and new beams cannot be allocated to the same beamlet number before killing the process.
NOTICE: If several pointing directions are combined in the same command, a duration must be defined for each of them. The directions will be then used one-by-one, changing to the next direction in the list after the duration of the previous one has ended.

`--anadir=longitude,latitude,type[,duration]` Pointing direction of the analogue HBA tile beams, in any of the coordinate systems understood by LOFAR (see Section 3.2.1). The pointing direction is updated with regular intervals, whose length is defined in the configuration file `/opt/lofar/etc/BeamServer.conf`. The values `longitude,latitude` are either longitudes and latitudes in their usual meaning, or azimuths and elevations, depending on the coordinate system. `type` is the name of the coordinate system. The optional `duration` is lifetime of the beam in seconds.
Examples: `--anadir=0,1.570796,AZELGEO` `--anadir=5.2336865,0.71094095,J2000`
NOTICE: Make sure that the digital station beam direction(s) given with the `--digdir` option fall inside the analogue tile beam. Tile beam width is approximately 30 degrees at 150 MHz, and it is inversely proportional to frequency.
NOTICE: If `duration` is given, the data flow will stop after the given time, but the `beamctl` process will continue running, and new beams cannot be allocated to the same beamlet number before killing the process
NOTICE: If several pointing directions are combined in the same command, a duration must be defined for each of them. The directions will be then used one-by-one, changing to the next direction in the list after the duration of the previous one has ended.

`--calinfo` Print information of available station calibration tables.

`--help` Print help message.

A.3 rspctl

Most functionality of the RSP boards, RCUs, and HBA tiles is controlled via the `rspctl` command. Note that much of its functionality is accessible via the `beamctl` command. If possible, the `beamctl` command should be used instead of `rspctl`, because it contains certain security checks that are not included in the `rspctl` command.

If the user wants to control the RSP without forming beams, it is advisable to switch to software level 2. At higher levels the beam server is running and may reset user-defined settings.

`rspctl` accepts several options, which are used as follows:

`--help` Print help message

`--select=<set>` Apply the command to selected RCUs only. Can be combined with most of the other options.
Examples: `--select=0:99`, `--select=0,1,4,5,16:29`

`--rcu` Show current RCU control settings.

`--rcu=0x00000000` Set the RCU control registers. The settings can be accessed via other `rspctl` options as well. For more information of the register settings, see `rspctl` help message. The meaning of the 24 bits are explained in more detail in [13, Section 3.8].

`--rcumode=[0...7]` Query or set the receiver mode. The possible values are given in the `rspctl` help message:

```
0 = OFF
1 = LBL 10MHz HPF 0x00017900
2 = LBL 30MHz HPF 0x00057900
3 = LBH 10MHz HPF 0x00037A00
4 = LBH 30MHz HPF 0x00077A00
5 = HB 110-190MHz 0x0007A400
6 = HB 170-230MHz 0x00079400
7 = HB 210-270MHz 0x00078400
```

See also the `--rcumode` option of `beamctl` in Appendix A.2.
NOTICE: The option should not be used for switching to receiver modes 5, 6, and 7. Use the script `startuphba.sh 5|6|7` instead.
NOTICE: Antenna cables can be safely (dis)connected in mode 0.

`--mode=[0..7]` Set rcumode is a specific mode. Enable (or disable) input from RCU's. Enable (or disable) spectral inversion. Set the hbadelays to default value 253 in HBA mode.

`--bitmode=[4|8|16]` Set or get the number of bits per sample. This must be set before the correct bitmode can be used with beamctl A.2.

`--rcuprsg=[0|1]` Turn pseudo-random number generator of a RCU off or on. Default `1` (on).
NOTICE: This setting overrules previous `rcumode` settings.

`--rcureset=[0|1]` Take RCU out of reset or hold RCU on reset. Default `1` (hold on reset).

`--rcuattenuation=[0...31]` Query or set the RCU attenuation settings. One step is 0.25 dB (1.0 dB in older RCU-II design).

`--rcudelay=[0...127]` Set the delay of a RCU. One step is 5 ns or 6.25 ns, depending on sample clock frequency (see `--clock`).

`--rcuenable=[0|1]` Disable or enable RCUs, i.e. switch data stream from RCUs to RSP boards on / off. Default `1` (enable).

`--specinv=[0|1]` Disable or enable spectral inversion. Default `1` (enable).

`--weights` Get weights of RCUs as complex values.
NOTICE: The unit used here is different from the unit used in other weight queries and settings. 16384 in this query is 1 in the following three options.

`--weights=value.re[,value.im ]` Set RCU weights as complex values. `value.re` and `value.im` should be floating point values in the range (-1,1].

`--aweights` Get weights of RCUs as amplitude and angle (in degrees).

`--aweights=amplitude[,angle]` Set RCU weights as amplitude and angle (in degrees).

`--subbands=[<set>]` Query or set the receiver subbands. The output for even-numbered RCUs is the true subband multiplied by two. The output for odd-numbered RCUs is the true subband multiplied by two, and the result incremented by 1. When setting the subbands, the true subband numbers are used. If `<set>` is shorter than 244, the remaining subbands are left unchanged.
Examples: `--subbands=100:199`, `--subbands=100:109,200,201`

`--xcsubband=[<int>]` Query or set the subband used for array covariances (crosslet statistics, visibility). The query prints four copies of the same value for each RCU. The output for even-numbered RCUs is the true subband multiplied by two. The output for odd-numbered RCUs is the true subband multiplied by two, and the result incremented by 1. When setting the subband, true subband numbers are used.

`--wg` Query settings of RSP board waveform generators.
NOTICE: This setting overrules previous `rcumode` settings.

`--wg=freq[--phase=... --amplitude=...]` Set waveform generator settings. Frequency is the frequency offset from the lower receiver band edge in Hz, phase in radians in the range $(0, 2\pi]$, amplitude is in range $(0, 1)$. All values are floats.
Example: `--wg=10000000 --phase=1.0 --amplitude=0.5`

`--version` Print version information.

`--status` Print status information of RSP boards.

`--tdstatus` Print status information of TDS (time distribution) boards.

`--spustatus` Print status of SPU (power distribution) boards.

`--realdelays=[<list>]` Query or set the 16 installed delays of HBA tile analogue beamformers. This option queries the values from the tiles themselves. For instructions of setting the delays, see `--hbadelays`.

`--regstate` Show update status of all registers once every second. The registers are up to date when all output lines consist only of the characters '*' and '.'.
NOTICE: Registers are updated every time the software level is changed from smaller than two to two or above, or when the sample clock frequency is changed. Do not give any other commands before the registers are up to date.

`--latency` Show latency of ring and all crosslet and beamforming lanes. "ring" refers to latency in a RSP board internal LVDS ring, whose latency is used as reference and should be zero.

`--statistics=[subband|beamlet --duration=<sec> --integration=<sec> --directory=<directory>]` Plot or write to files subband / beamlet statistics. See sections 4.1 and 4.2 for more details.

`--xcstatistics[ --duration=<sec> --integration=<sec> --directory=<directory>]` Plot or write to file array covariances (crosslet statistics). See Section 4.3 for details.

`--clock=[0|160|200]` Query or set the sample clock frequency. `0` is off, the other two are frequencies in MHz.
NOTICE: A change in the clock frequency causes a register update, and one needs to wait for it to complete before giving further commands. Use `rspctl --regstate` to verify that the registers are up to date.

`--rspclear` Clear FPGA registers on RSP boards.

`--hbadelays=[<list>]` Get or set the 16 delays of HBA analogue beamformers. This option does not query the values from the tiles, and its output may thus be different from the output of the `--realdelays` option. The delays are set in steps of 0.5 ns in the range 0 – 15.5 ns. The option also allows one to switch off LNAs of individual antenna elements (30 dB suppression of the signal), or completely switch off antenna elements. Functions and values of each input bit are listed in Table 8. The input values are given as decimal integers. Some example values are given in Table 9.
Example: `--hbadelays=2,2,2,2,2,2,2,2,2,2,2,2,2,2,2,2 --select=0,1`
NOTICE: This option is usually combined with the `--select` option. The same delay value is copied for both X and Y -polarisations of each antenna element.

`--tbbmode=[transient | subbands,<set>]` Query or set the transient buffer board recording mode. With `--tbbmode=subbands`, one needs to selected the RCUs as well.
Examples: `--tbbmode=transient`, `--tbbmode=subbands,0:242`
NOTICE: Maximum 487 subbands can be transferred to the TBB. The system does not make any sanity checks about the length of `<set>`.

Table 8: Meanings of HBA delay setting bits.

	Function	Value
(MSB) Bit 7	RF on / off	1 = on
Bit 6	8 ns	1 = 8 ns
Bit 5	4 ns	1 = 4 ns
Bit 4	2 ns	1 = 2 ns
Bit 3	1 ns	1 = 1 ns
Bit 2	0.5 ns	1 = 0.5 ns
Bit 1	LNA on/off	1 = off
(LSB) Bit 0	LED on/off	1 = on

Table 9: Examples of using the HBA delay settings.

Input	Function
0	0 ns delay, LNA on, RF off (= ca. 30 dB suppression of the signal)
2	LNA off, RF off, (completely off)
3	LNA off, RF off, LED on
128	LNA on, RF on, 0 ns delay
252	LNA on, RF on, 15.5 ns delay

`--splitter=[0|1]` Set or get the status of serdes splitter. `0` is off, `1` is on. This command has effect only at core stations, where the HBA array is divided into HBA0 and HBA1 arrays. It splits the beamforming rings in two parts, thus allowing 488 beamlets to be formed, but each of them using only half the number of antenna elements.
NOTICE: The splitter must be manually set on when doing beamforming with HBA0 and HBA1 separately, i.e. when using `beamctl --antennaset=HBA_DUAL`. When the splitter is on, beamlet numbers 1000 – 1243 produced from the HBA1 array are output from different RSP boards than the usual beamlets 0 – 243. These boards are defined in `/opt/lofar/etc/RSPDriver.conf`.

`--datastream=[0|1|2|3]` Query the status or switch on/off the data stream to CEP. Default `1` (on). Option `1` starts the data stream only from ring 1, option `2` only from ring 2, and option `3` from both rings simultaneously.
NOTICE: `rspctl --datastream=2|3` are intended for use only at core stations, when combined with `rspctl --splitter=1`.
NOTICE: The command will always output the status of both rings as follows:
`Datastream to CEP switched 0:on 1:off`

`--swapxy=[0|1]` Set or get the status of xy connector swap. `0` is off, `1` is on. Default `1` (on).
NOTICE: This option is usually combined with the `--select` option, because swapping all x-connectors to y and vice versa may not be useful.
NOTICE: This option has no impact on the array covariance matrix input order 4.3. The order is determined by the fysical RCU number 0..95 [0..191]. The order is XYXY..XY.

A.4 tbbctl

The TBB boards are controlled with the `tbbctl` command, whose options are used as follows:

`--select` Apply the command to selected RCUs only. Can be combined with most options. NOTICE: This option is used only together with other options, it has no meaning when used alone.

`--alloc` Allocate memory for the RCUs. All available memory in each TBB is divided in between the selected RCUs. Reducing the number of RCUs will thus lead to larger buffers.

`--free` Free memory.

`--record` Start recording.

`--stop` Stop recording.

`--mode=[transient|subbands]` Set receive mode to configure UDP/IP header for CEP data transport. NOTICE: This command does not set the data recording mode, it sets only the page header. Use `rspctl --tbbmode` to set the recording mode.

`--storage=node` Set storage node for CEP data transport. `node` should be one of the storage node names listed in the file `/opt/lofar/etc/StaticMetaData/Storage+MAC.dat`.

`--read=rcunr,secondstime,sampletime,prepages,postpages` Read data from TBB memory and transport it to CEP. The arguments are RCU number, full second, sample count from full second, memory page offset backwards from the given sample, and memory page offset forwards from the given sample.
NOTICE: Recording must be stopped with `tbbctl --stop` before giving this command.

`--readall=pages` Read the latest `pages` recorded pages from all (selected) RCUs, and transport the data to CEP.
NOTICE: This command can be given also while receiving data. The recording will be automatically stopped, as well as restarted after the reading.

`--stopcep` Stop sending data to CEP.

`--cepdelay=delay` Set time delay between subsequent CEP frames. One delay unit is 5 μs.

`--arp` Send one arp message.

`--arpmode=<0|1>` Set arp mode, `0` is manual, `1` is automatic (1 message every 42 seconds).

`--settings` List trigger settings.

©ASTRON 2018

`--release` Release triggers. Releasing a trigger means that the system starts looking for triggering pulses, and sends trigger messages after detecting such a pulse.

`--generate` Generate a trigger signal manually from LCU command line.

`--setup=level,start,stop,filter,window,mode` Set up a triggering system running on the TBB boards. The system calculates mean and standard deviation of the incoming signal. If an incoming sample differs at least `level` standard deviations from the mean value, a counter is incremented by four. Otherwise the counter is decremented by one. A counter value above `start` will be considered as a triggering pulse. Another counter counts consecutive samples below the trigger level, and reaching the value `stop` is considered as the end of the triggering pulse. A trigger signal is transmitted when the end condition is meat. `filter` selects if the incoming data are filtered before the inspection, `0` enables the filter, `1` disables it. `window` is an integer from range (0,8), it selects length of the integration window used for the mean and variance calculations. The window length is $2^{12+\texttt{window}} \cdot 5 \cdot 10^{-9}$ s if 200 MHz sample clock is used, and $2^{12+\texttt{window}} \cdot 6.25 \cdot 10^{-9}$ s if 160 MHz sample clock is used. `mode` is selection between `one_shot` and `continues` modes, which are selected with numbers `0` and `1`, respectively. In one shot mode only one trigger signal per RCU will be sent, in continues mode the trigger is released after a detection. See [6] and [4] for more information.

`--coef=f00,f01,f02,f03,f10,f11,f12,f13` Set filter coefficients for the triggering system. See [6] for details.

`--triginfo=rcu` Get trigger info for selected RCU.

`--listen=[one_shot|continues]` Set LCU to listen for triggers.
NOTICE: This option only prints information about received trigger signals, it does not stop the recording after receiving a trigger signal.

`--rcuinfo` List info of the RCUs, for which memory has been allocated.

`--version` Print version information.

`--status` Print status of TBB boards.

`--size` Get installed memory sizes of the TBB boards.

`--readpage=rcunr,startpage,npages` Read `npages` of TBB data from RCU `rcunr` to a file, starting from page `startpage`.
NOTICE: The `--readpage` option is developed only for testing purposes, it writes the data very slowly.

`--imageinfo=board` Print information of FPGA images on a board.

`--config=imagenr` Reconfigure boards with image number `imagenr`.

`--clear`	Clear a whole board.
`--reset`	Reset to factory images.
`--help`	Print help message.

B LOFAR station configuration files

A LOFAR station has a number of configuration files ,which are stored at `/opt/lofar/etc`. The following is a list of configuration files which may be relevant for a single-station user.

`AntennaField.conf` Positions of the antenna elements. The file contains positions of antenna array centres and antenna element offsets from the corresponding array centres in cartesian ITRS coordinates, together with a rotation matrix for transforming the antenna element offsets between ITRS and station coordinates. Station coordinate system is a cartesian system with horizontal q-axis pointed along the bisector of X- and Y-dipoles, horizontal p-axis perpendicular to the q axis, and r-axis pointing upwards [3]. The p-axis direction is such that (p, q, r) is a right-handed coordinate system.

`AntennaSets.conf` Definitions of antenna sets. The physical antennas are referred to in the same order as they are given in the file `AntennaField.conf`. Instructions for defining antenna sets are given in the file header.

`CableAttenuation.conf` Attenuations in coaxial cables of different lengths at each frequency band. NOTICE: An attenuation must be defined for all cable lengths in `CableDelays.conf`.

`iHBADeltas.conf` Positions of the individual antenna elements inside the HBA tiles.

`HBADelays.conf` Possible delay values of the analogue HBA tile beamformer.

`CableDelays.conf` Group delays in the coaxial cables connecting the antenna elements and HBA tiles to the RCUs. Average values by default, but can be replaced with measured values if available. Note that the file contains columns for both physical length and delay of the cables. The RSPDriver requires that the physical length is one of the values for which an attenuation is defined in `Attenuation.conf`, otherwise the RSPDriver will not start.

`swlevel.conf` Definitions of the processes to be started at each software level.

`RSPDriver.conf` RSP driver configuration. Contains settings for CEP data packet source and destination addresses, number of time-slices and subbands per packet, selection of output RSP boards, etc.

`BeamServer.conf` Beam server configuration. Contains beam direction update intervals, names of antenna field configuration files, and settings for communication between beam server and RSP driver.

`SourceCatalog.conf` Radio source catalogue for station calibration.

`CalServer.conf` Calibration server settings, e.g. settings for writing calibration ACC data to file [See 18].

`RemoteStation.conf` Description of station hardware, number of signal processing boards etc.

`TBBDriver.conf` TBB driver settings, e.g. destination addresses for UDP packets sent from TBB to CEP.

`DipoleModel.conf` Model of the LBA dipole for station calibration.

`StaticMetaData` A directory that contains configuration files of all lofar stations, calibration tables, etc.

`PPSdelays.conf` Additional delay between rising edge of PPS and RSP Antenna Processors (AP). The delay resolution is in 63 steps of 78 fs. At the moment all 48 APs for the dutch and 96 APs for the international stations are fine-tuned for optimum alignment of the sampling clock.

©ASTRON 2018

C CEP data recording with Wireshark

A LOFAR station outputs CEP data from at least four – in core stations even from eight – RSP boards, and from all of its TBB boards. In order to record beamformed voltage level data, one needs to capture the data packets transmitted from the RSP boards. Though there is a possibility to store TBB data to files, the CEP output is the primary output route for TBB data as well, and recording substantial amounts of TBB data at single-station level is practical only by capturing the CEP data packets.

A system for large-scale CEP data recording would most probably use more sophisticated methods than those used in this example. However, this example does not require anything else than a laptop that is connected to a data output port of a RSP board or a TBB. Anyone with access to a LOFAR station should thus be able to follow the instructions and to record sample CEP data.

This example shows how to capture RSP or TBB data by connecting a receiving computer directly to the data output ports of the RSP boards or the TBBs. Connecting directly to the boards is the simplest way to get access to the CEP data, but it is obviously not very good solution for larger scale data recording, because one has to manually unplug the original data output cables and to plug in the receiver computer. A more sophisticated solution could be e.g. to connect the receiving computer via a network switch, which is the solution used by the ARTEMIS system http://www.oerc.ox.ac.uk/research/artemis. However, the data packets are the same regardless of the connection point, and the simplest possible data recording method is thus a natural starting point e.g. when designing more sophisticated recording systems.

Notice that the CEP data output uses jumbo ethernet frames (9000 bytes payload), whose use one may need to manually enable for the NIC of the data receiving computer. The CEP data packets will be ignored by the receiving computer if the jumbo frames are not supported. If necessary, beamlet data packet size can be decreased by adjusting the number of transmitted beamlets (see Section 5.6).

In this example Wireshark http://www.wireshark.org is used for capturing the data packets. The software is used as a demonstration tool, because it is well known, easy to use, and available for most common computer platforms (Windows, OS X, Linux, ...). Wireshark also supports packet capture in "promiscuous" mode, in which all packets arriving to a network connector are recorded, regardless of their destination address. Wireshark thus allows data recording without making changes to the LOFAR station configuration files. At the time of writing this document, the software and its user guide were available at http://www.wireshark.org. Very useful information for Wireshark users is available also in http://wiki.wireshark.org/

C.1 Connecting a receiving computer

The receiving computer needs to be connected to one of the output RSP boards (if beamlet data are being recorded) or to a TBB (if TBB data are being recorded). The output RSP boards are normally the boards 0 – 3, which are found from subrack 0 of a station [see 12]. If this would not be the case,

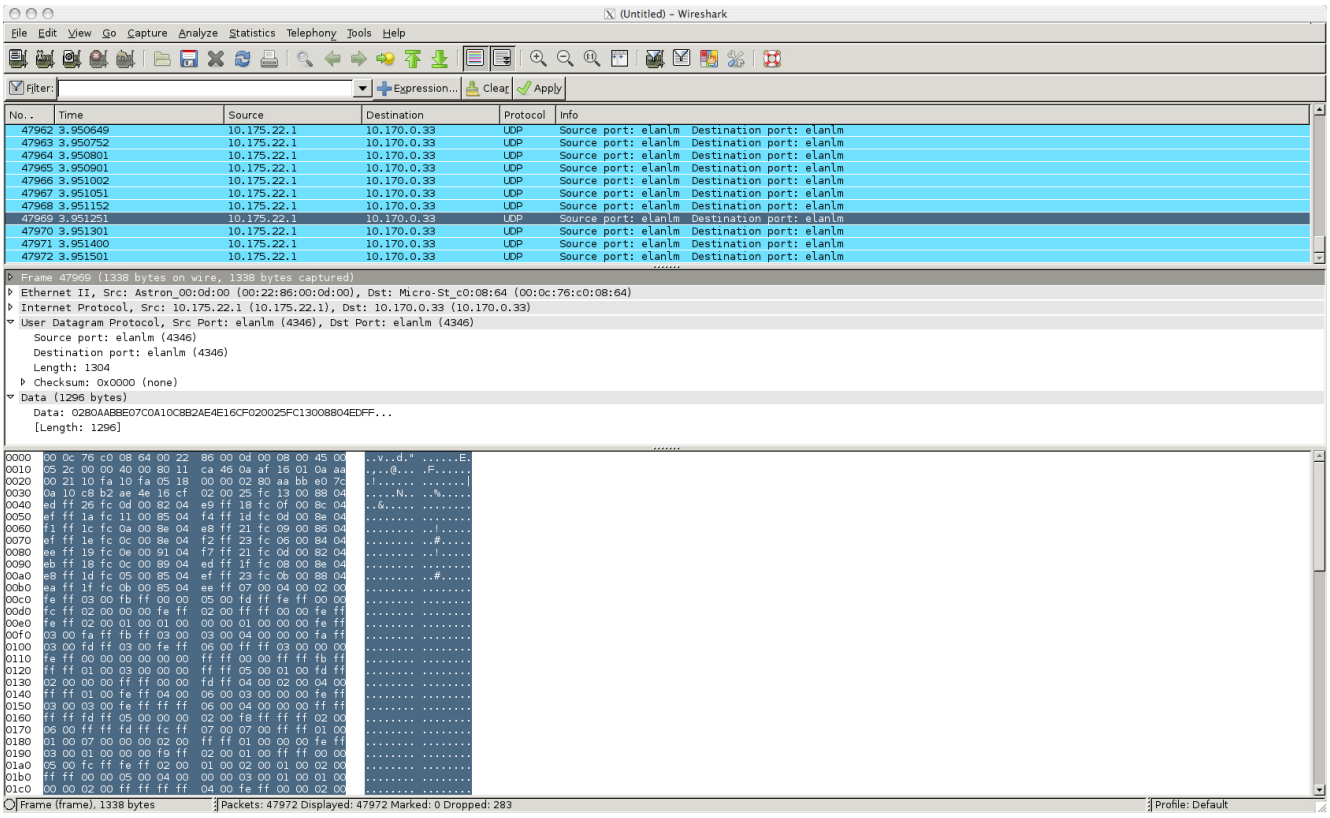

Figure 5: A screen shot of a Wireshark GUI after capturing CEP beamlet data. The packet source IP address is 10.175.22.1 and destination address is 10.170.0.33. The data are being transmitted from port 4346 to the same port number. Also the source and destination MAC addresses, network protocol, etc. are given.

the output RSP boards can be found from the configuration file `/opt/lofar/etc/RSPDriver.conf`. The same file contains also the additional output board numbers of core stations, as well as IP and MAC addresses written to the data packet headers. Each RSP board outputs 61 beamlets, with beamlets 0 – 60 output from RSP board 0, beamlets 61 – 121 from RSP board 1, etc. All TBBs output data, with data from RCUs 0 – 15 output from TBB 0, RCUs 16 – 31 from TBB 1, etc. The CEP data output ports are clearly marked on both RSP boards and TBBs.

Each data output port should have a cable plugged in, which normally connects the port to the main data switch of the station. One needs to unplug this cable and to connect the receiver computer to the data output port. Remember to plug in the original cable after recording sample data!

C.2 Wireshark GUI

The graphical user interface of wireshark is rather self-explanatory. In order to record a few seconds of data, one only needs to select the correct network interface and start recording. The recording can then

be stopped with a "stop" button, and the whole recording can be stored in a file. For more advanced use several options are available. One can e.g. record only packets arriving from a given address, which may be useful in some cases. When recording larger amounts of data, one can configure wireshark to write the data directly to files of given size, etc.

An example of the GUI after capturing beamlet CEP data is given in Figure 5. The output indicates that packets are being transmitted from IP 10.175.22.1 with destination IP 10.170.0.33. The data are being transmitted from the elanlm port 4346 to the corresponding port in the remote machine, which are the default ports for CEP data. The lowest panel in the window shows the data values in the packet. If the beam server is not producing any data, there will be non-zero values only in the packet header, which fits inside the first four lines of the data window. In this example all rows contain non-zero data, indicating that the beam server is running and at least one beam has been allocated.

The captured packets can be stored in files for later use. Wireshark writes the data in libcap file format (see Section C.4). Inside the libcap files the original ethernet frames are stored as such. For the ethernet frame format, see Section 5.6. The software allows recorded packets and their contents to be filtered, which may be practical when inspecting contents of the captured data. Instructions for filtering can be found from the Wireshark user guide http://www.wireshark.org/docs/ and wiki http://wiki.wireshark.org/.

C.3 tshark

tshark is a command-line interface for wireshark. tshark man pages (`man tshark`) and help message (`tshark -h`) provide all necessary information for data collection. Below is a simple example of its use:

```
# list available interfaces
> tshark -D
1. en0
2. fw0
3. en1
4. lo0

# capture data from interface en1 and save the packets in a libcap-file "data" (stop with ctrl-c)
> tshark -i en1 -w data
Capturing on en1
19 ^C

# list captured packages
> tshark -r data
```

Table 10: Layout of libcap global header. All fields are unsigned integers.

Octets	Contents
0 – 3	Magic number
4 – 5	Libcap version number (major)
6 – 7	Libcap version number (minor)
8 – 11	GMT to local time correction
12 – 15	Accuracy of timestamp
16 – 19	Snapshot length (maximum length of a captured packets, in octets)
20 – 24	Data link type

Table 11: Layout of libcap packet header. All fields are unsigned integers.

Octets	Contents
0 – 3	Timestamp seconds
4 – 7	Timestamp microseconds
8 – 11	Number of data bytes actually saved in the file
12 – 15	Original number of data bytes in the packet

If larger amounts of data are recorded, it may be practical to divide the data in several files:

```
# switch to a new file after 10 kB  and stop after three files
> tshark -i en1 -w data -b filesize:10 -a files:3
Capturing on en1
241
```

C.4 Libcap file format

A libcap file consists of a global header, packet headers, and packet data. Each file begins with a global libcap header, which is immediately followed by the libcap packet header of the first captured packet. The libcap packet header is followed by the recorded ethernet frame, including the ethernet header but excluding the frame preamble and start of frame delimiter. In other words, the packet data begins from octet 0 in table 6, the destination MAC address.

The global libcap file header is 24 octets long, and it contains the information listed in Table 10. The magic number in the beginning of the header is for endianness checking, if the reading system reads `0xa1b2c3d4` it uses the same endianness that the file has, if it reads `0xd4c3b2a1` the bytes need to be swapped. The libcap packet header is 16 octets long, and it contains the information listed in Table 11. Further information about the libcap file and hader formats is available in Wireshark wiki http://wiki.wireshark.org/Development/LibpcapFileFormat.

A libcap file contains a set of captured packets in the order global libcap header, packet 0 libcap header, packet 0 data, packet 1 libcap header, packet 1 data, packet 2 libcap header, ...