ASTRON
Netherlands Institute for Radio Astronomy

Apertif and Arts Firmware Quantization Model

	Organisatie / Organization	Datum / Date
Auteur(s) / Author(s): Eric Kooistra	ASTRON	2019
Controle / Checked: Gijs Schoonderbeek	ASTRON	
Goedkeuring / Approval: Agnes Mika	ASTRON	
Autorisatie / Authorisation: **Handtekening / Signature** Agnes Mika	ASTRON	

Distribution list:

Group:	Others:
Apertif project DESP	

Document history:

Revision	Date	Author	Modification / Change
0.1	25 Feb 2019	E. Kooistra	First draft. Open issues marked with ???

Table of contents:

References:

[1] “Detailed design of the Arts FPGA Beamformer”, ASTRON-SP-062, 2017, E. Kooistra
[2] “Analysis of tied-array beamforming for Arts”, ASTRON-MEM-199, 2018, S.J. Wijnholds
[3] “Understanding Digital Signal Processing”, 2010, R.G. Lyons
[4] “Apertif Firmware Verification and Maintenance”, ASTRON-RP-010887, 2019, E. Kooistra
[5] apertif_arts_firmware_model.py
https://svn.astron.nl/UniBoard_FP7/RadioHDL/trunk/applications/apertif/matlab/apertif_arts_firmware_model.py
[6] Apertif application control classes and commands:
https://svn.astron.nl/UniBoard_FP7/UniBoard/trunk/Software/python/peripherals/pi_apertif_system.py
[7] Apertif control at WSRT using UPE:
https://svn.astron.nl/UniBoard_FP7/RadioHDL/trunk/applications/apertif/doc/apertif_wsrt_useful_commands_erko.txt

The latest versions of the ASTRON documents are available at:

https://svn.astron.nl/UniBoard_FP7/RadioHDL/trunk/applications/arts/doc/
https://svn.astron.nl/UniBoard_FP7/RadioHDL/trunk/applications/apertif/doc/

The UniBoard_FP7 SVN repository can be accessed using the read only SVN account:

Username = uniboard_fp7-guest
Password = uniboard_fp7-guest

Terminology:

ADC	Analogue to Digital Converter
ADUH	Analogue Digital Unit Handler
Apertif	APERture Tile In Focus
Arts	Apertif Radio Transient System
beam	Group of beamlets or beamlet-channels that point in the same direction
beamlet	Beam formed subband by the Apertif BF, a small beam spanning one subband
BF	BeamFormer
BN	Back Node (FPGA on UniBoard)
BSN	Block Sequence Number (= a timestamp)
BST	Beamlet Statistics
CB	Compound Beam, formed at dish level over the FPA by the Apertif BF
channel	Unit frequency band within a beamlet
DB	Data Buffer
DSP	Digital Signal Processing
DW	Data Writer
FIR	Finite Impulse Response filter
FPGA	Field Programmable Gate Array
F_{chan_x}	Channel Filterbank (fine channels)
FN	Front Node (FPGA on UniBoard)
FoV	Field of View
FPA	Focal Plane Array (= PAF)
F_{sub}	Subband Filterbank (coarse channels)
IAB	Incoherent Array Beam
IQUV	Stokes intensity and polarization powers
LSbit	Least Significant bit
MAC	Monitoring and Control
PAF	Phase Array Feed (= FPA)
rms	Root Mean Square (= std() when mean = 0)
RFI	Radio Frequency Interference
SC	Science Case (of Arts)
SNR	Signal to Noise Ratio
SST	Subband Statistics
std	Standard deviation (sigma)
subband	Coarse channel frequency band, unit output of the Apertif BF filterbank
TAB	Tied Array Beam
TP	Telescope Path (one polarization from one dish)
X, XC	Correlator
WG	Waveform Generator

Definitions:

For parameter definitions see [1] and [6].

1 Introduction

1.1 Purpose

This document describes the fixed point quantization model of the digital signal processing that is done in the firmware of the FPGAs in Apertif and Arts [1]. The model is called ***apertif_arts_firmware_model.py*** [5] and is written in Python. The firmware provides several monitoring points that can be read via the monitoring and control (MAC) interface to verify that the (internal) signal levels of data agree with the model.

The purpose of the model is to understand the signal levels in the data path processing and to show that the internal requantization in the firmware implementation does not increase the system noise while providing the maximum dynamic range [2]. The fixed point signal representations are shown by the green bars in Figure 5 for Apertif BF and X, in Figure 6 for Arts SC1 TAB, in Figure 7 for Arts SC4 TAB and in Figure 8 for Arts SC4 IAB and by the simulation log results in section 4.2.

The quantization levels have been verified on hardware. For Apertif part of the results are reported in section 5 and more in [4].

1.2 Scope

This document covers both fixed point signal levels in both the firmware for Apertif and for Arts. For more details on how the model works and is used also please read the help in [5].

2 System overview

2.1 Block diagrams

The detailed design document of the Arts firmware in [1] provides a top level design description of the Apertif beamformer (BF) and Apertif Correlator (XC) and the Arts Array Beamformer (IAB, TAB). The Apertif BF shown in Figure 1 captures the ADC data and forms compound beams (CB) that are output as beamlets to both Apertif X and Arts. The Apertif X correlates fine channels to create channel visibilities and is shown in Figure 2. There are N_{chan_x} = 64 fine channels per subband. Arts consists of several science cases. Arts SC1 for pulsar timing is shown in Figure 3 operates on a voltage TAB that is directly created from the beamlets. Arts SC3 and SC4 for transient search are shown in Figure 4 and create power IAB or power TAB per Arts channel. There are N_{chan_bf} = 4 Arts channels per subband. The white blocks in the figures indicate functions in FPGA firmware, the yellow blocks show the external input and output components.

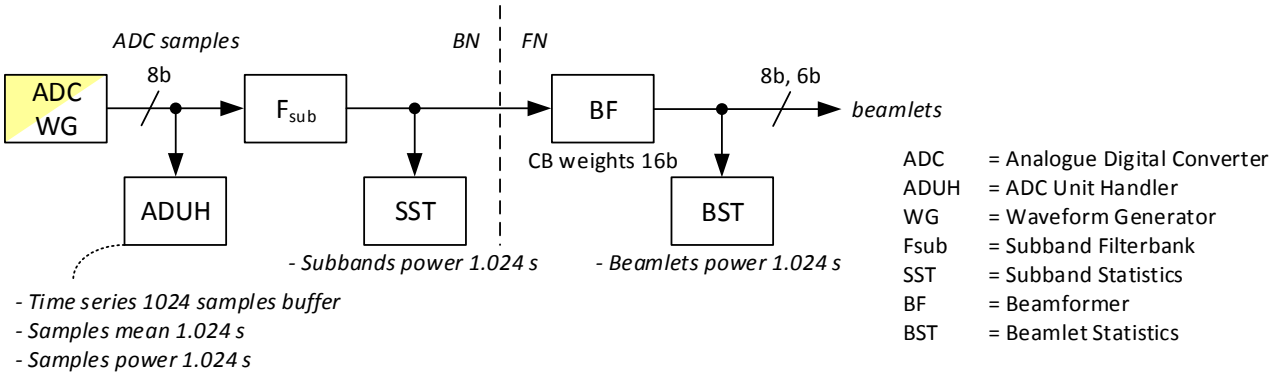

Figure 1: Apertif Compound Beamformer (BF) at dish

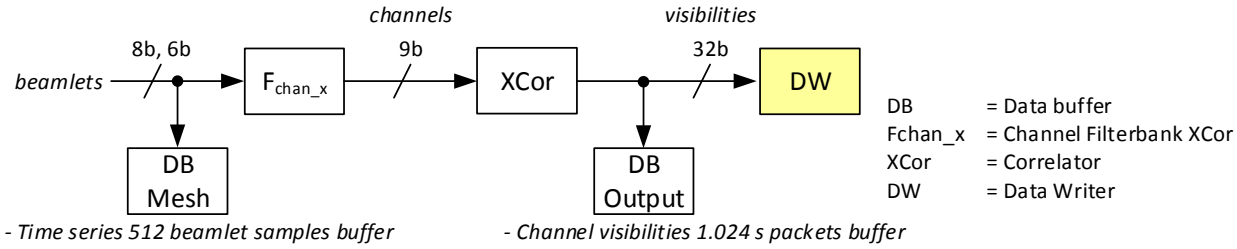

Figure 2: Apertif Correlator at central

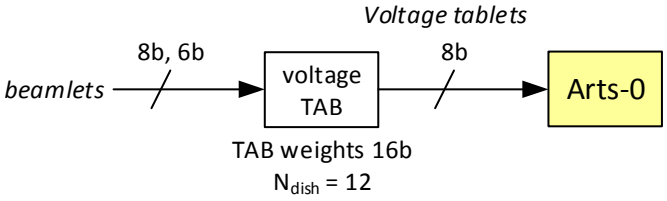

Figure 3: Arts SC1 Pulsar Timing voltage TAB at central

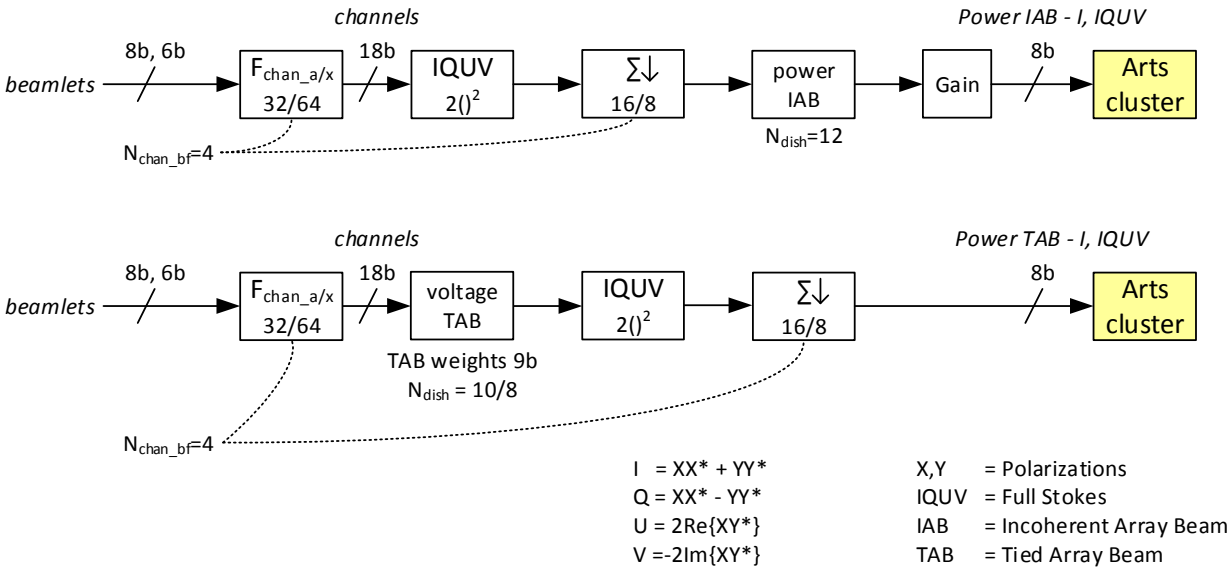

Figure 4: Arts SC3,SC4 Transient Search power IAB and power TAB at central

Note: The Arts images use the same beamlet input section as the Apertif X. Therefore Arts SC1 and Arts SC4 can also capture beamlets in the DB mesh data buffer, provided that this DB mesh peripheral was enabled in these images at compile time.

2.2 Data widths at external interfaces

The data widths at the external interfaces are fixed [6]:

- W_{adc} = 8 bit at the ADC
- $W_{beamlet}$ = 6 bit between dish and central
- W_{vis} = 32 bit for the channel visibilities output to the DW
- $W_{channel\_x}$ = 9 bit for the fine channels that Apertif X correlates and outputs to commensal Arts SC3
- W_{volt} = 8 bit for the voltage TAB of Arts SC1
- W_{pow} = 8 bit for the Stokes powers in the IAB and power TAB of Arts SC3 and SC4.

2.3 Data control and monitoring points

The BN in the Apertif BF contain waveform generators (WG) that can replace the ADC data and generate a sinus signal with amplitude, phase and on average any frequency between 0 and $f_s/2$ = 800MHz / 2 = 400 MHz. Using the BSN scheduler in the BN it is possible to start multiple WG on different dishes at the same time. In the data path processing at the dish the signal can be monitored via the MAC interface at the ADUH, the SST and the BST, see Figure 1. At the central XC a snapshot of the input beamlets can be captured in the DB mesh, see Figure 2. The output visibilities of the XC are further processed on the external DW, but a snapshot of N_{chan_x} = 64 channel visibility packets, so all channels for one beamlet subband, can be captured in the DB output in Figure 2. The DB output can capture data from the sync start of a 1.024s interval or starting at any sample within a sync interval. For the Arts applications there are no dedicated data monitoring points in the firmware, but the Stokes I and IQUV data of the IAB and TAB can be logged in the Arts pipeline on Arts0 for SC1 and the Arts cluster for SC3, SC4.

3 Fixed point data model

3.1 Introduction

3.1.1 System noise and weak astronomical signal

The weak astronomical signal is buried in the system noise with a signal to noise ratio (SNR) <<< 1. Assume only the system noise is relevant to determine the proper internal fixed point signal levels, because fine channels will still have SNR << 1. This means that the quantization in the data path firmware should preserve sufficient bits to represent the input system noise. The system noise is asynchronous per dish. The weak sources will be received synchronously at the dishes, but only become apparent during correlation (Apertif XC) or folding (Arts). This implies that $W_{beamlet}$ and $W_{channel\_x}$ can be dimensioned to fit e.g. 3 sigma of the system noise, so about extra $\log_2(3) \approx 1.6$ bits.

At the ADC the system noise is sampled such that the sigma is represented by about 8, so 3 bits. Two more bits can represent four sigma, because $\log_2(4) = 2$. The ADC in Apertif has $W_{adc} = 8$ bits and the extra bits provide more dynamic range to avoid overflow and clipping due to RFI. At the ADC clipping must be avoided, because the non-linear clipping disturbs all subbands. Similar the internal data path processing is dimensioned to have sufficient bits to avoid internal overflow. RFI can be modelled by a WG signal. The input PFB must have sufficient dynamic range to full scale RFI in W_{adc}, to avoid interference in all subbands.

The external interface at the dish BF output to the central UniBoards forms a bottleneck, because the beamlets are represented by $W_{beamlet} = 6$ bits. The WG provides a coherent source and even with a small amplitude the coherent WG input will cause clipping at the 6 bit beamlets and at the $W_{channel\_x} = 9$ bit channels. The assumption is that strong RFI is limited to single subbands or fine channels that will then clip. The clipped subbands and fine channels are not useful for astronomical measurements. However, the other subbands and fine channels that did not clip still carry a linear representation of the weak astronomical source, that can be detected in the correlator or in the Arts pipeline.

3.1.2 Quantization noise

The quantization noise power is $q^2/12$, so the rms quantization noise $\sigma_q = 0.29\ q$, where q is the unit quantization step of 1 LSbit [3]. If the rms of the system noise is represented by a few quantization levels q or LSbits, then the effect of the quantization noise on the weak astronomical signal is negligible compared to the effect of the system noise. With $\sigma_{sys} = nq$ and $\sigma_q = q/\mathrm{sqrt}(12)$ and $\mathrm{sqrt}(1+\varepsilon) \approx 1 + \varepsilon/2$ for small ε, the total rms noise becomes [2]:

Equation 1
$$\sigma_{tot} = \sqrt{\sigma_{sys}^2 + \sigma_q^2} = q\sqrt{n^2 + \frac{1}{12}} = qn\sqrt{1 + \frac{1}{12\,n^2}} \approx \sigma_{sys}\left(1 + \frac{1}{24\,n^2}\right)$$

The ratio $\sigma_{tot} / \sigma_{sys} = 1 + 1/(24\ n^2)$, so for n = 2 the impact of the quantization noise on the sigma of the total noise is only about 1 %. With a sigma of e.g. 2 q, so $\log_2(2) = 1$ bit the maximum 3 sigma noise then adds about $\log_2(3) \approx 2$ bits and 1 sign bit, so then 4 bit will fit 99 % of the Gaussian noise. Extra bits are only needed to increase the dynamic range. Extra bit are not needed to improve the sensitivity, because the sensitivity is already preserved by internally representing the system noise with about the same number of bits as at the ADC input.

Note:
For LOFAR it seems necessary to represent the system noise by more LSbit, because this reduces impact of RFI via the nonlinear effect of quantification with only a few LSbits ????

3.1.3 Coherent and incoherent input

Model coherent ADC input using a Waveform Generator sinus and model incoherent ADC input using Gaussian white noise. The WG sinus is used to provide narrow band stimuli for a subband or a channel. It is easier to create a sinus than to create a narrow band noise signal. The WG in firmware, see Figure 1, can only model a coherent signal at a single frequency, so only in one subband, or one fine channel or one Arts BF channel. The WG is useful as a test signal to verify the internal signal levels within the data processing. The WG can also model strong RFI. The WG in the firmware cannot model the weak astronomical signal, because its minimal amplitude is >= 1 q and it cannot be applied in combination with system noise.

3.1.4 Signal power and signal amplitude

Some useful relations:

```
- power = rms**2
- mean powers = (rms voltages)**2
- rms**2 = std**2 + mean**2 = std**2 when mean = 0
- rms = std = A / sqrt(2) when mean is 0 → pi_apr.wgScale = sqrt(2) [6]

- power complex = power real + power imag
                = (rms real)**2 + (rms imag)**2
                = (A real/sqrt(2))**2 + (A imag/sqrt(2))**2
- power real = power imag = power complex / 2, when signal is random or periodic
- rms real = rms imag = rms complex / sqrt(2), when signal is random or periodic
  → pi_apr.rcScale = sqrt(2) = sqrt(nof_complex) [6]
- rms complex = sqrt(power complex)
- ampl real = ampl imag = sqrt(power complex / 2) * sqrt(2)
                        = sqrt(power complex)
                        = rms complex
                        = (rms real) * sqrt(2)
                        = (rms imag) * sqrt(2)
```

3.2 Apertif BF and Apertif X

3.2.1 Subband filterbank F_{sub}

A subband sample output of the F_{sub} in the Apertif BF needs to be represented by W_{fsub_gain} = 5 extra bits. For a WG input all power will appear in on subband, so with N_{fft} = 1024 real input samples and N_{sub} = 512 complex output samples this means that the gain from power per input sample to power at the specific complex subband sample is a factor N_{fft} = 1024. For Gaussian white noise all input power will appear in all subband samples, so a gain of 1. Hence the dynamic range of a subband increase by a factor N_{fft} in power and a factor sqrt(N_{fft}) in voltage. Therefore the subband samples need $W_{fsub\_gain} = \log_2(\sqrt{N_{fft}})$ = 5 extra bits to preserve the input sensitivity.

3.2.2 Compound beamformer

The CB beamformer in the Apertif BF sums the S = 64 inputs from the PAF to form a compound beam (CB). The shape of the compound beam is determined by the dish beam, which has a much smaller field of view (FoV) than the PAF element beam. The CB beamformer improves the shape of the CB compared to the shape of a single element CB, but it does not change the size of the FoV of the CB, because the size is determined by the dish. In terms of coherent versus incoherent input this means that also for incoherent input the compound beamformer acts coherently. Therefor the output of the compound beamformer has the same dynamic range as the input and does not need to be represented by extra bits.

The CB weights are W_{cb_weight} = 16 bit. The unit gain of the CB beamformer weights is $W_{bf\_unit\_gain}$ = -2. The firmware is coded such that for a CB weight of $2^{**}(W_{cb\_max\_weight} + W_{bf\_unit\_gain})$ = 8192 the beamlet output has

the same range as the subband input, so unit gain. Together the weights of a CB should be normalized to 8192. Somewhat larger or smaller CB weights can be applied to compensate for analogue gain differences per PAF element. Typically only 3 to 5 PAF elements are the main contributors to a CB, because these are in view of the dish beam for that particular CB pointing.

3.2.3 Fine channel filterbank F_{chan_x}

The fine channel filterbank F_{chan_x} in the Apertif X has N_{chan_x} = 64 complex input samples and also N_{chan_x} complex output samples. Hence the channel output of F_{chan_x} needs to be represented by $W_{fchan\_x\_gain}$ = $\log_2(\text{sqrt}(N_{chan\_x})$ = 3 extra bits to preserve the input sensitivity.

3.2.4 Correlator

The Apertif correlator (XC) correlates the channels between all dual pol dishes, so for $N_{pol} * N_{dish}$ = 2 * 12 = 24 telescope paths (TP). This results in $N_{tp} * (N_{tp} + 1)/2$ = 300 visibilities. The visibilties are integrated over $N_{int\_chan\_x}$ = 12500 channel samples in time for integration period t_{int_x} = 1.024 s. With $W_{channel\_x}$ = 9 bit channels this means that the visibility powers fit in $W_{channel\_x} + W_{channel\_x} + \log_2(N_{int\_chan\_x})$ = 9 + 9 + $\log_2$(12500) = 31.6 bits, so W_{vis} = 32 bit is sufficient to carry all values, without any extra requantization. For coherent input the integration yields a gain of $N_{int\_chan\_x}$, for incoherent input the gain is sqrt($N_{int\_chan\_x}$). Therefore the XC needs $\log_2(\text{sqrt}(N_{int\_chan\_x}))$ = $\log_2$(sqrt(12500)) = 6.8 extra bits.

3.2.5 Processing gain

Together the F_{sub}, F_{chan_x} and XCor provide a processing gain of 5 + 3 + 6.8 = 14.8 bits for the channel visibility data in Apertif X, or an SNR improvement of 20 * $\log_{10}$(2**14.8) = 89.1 dB. Hence for an output SNR of 1 the sigma of the weak coherent signal at the ADC input is then a factor 2**14.8 = 28.6e3 less than the system noise level, so completely buried in the system noise.

3.2.6 Fixed point data ranges

Figure 5 shows the quantization levels within the Apertif BF and the Apertif X relative to the input ADC level. The bold reference line marks the 1 quantization unit reference level of the ADC, below this line the levels are fractions of 1. The green bar at the input indicates a sigma level of 2 bits for ADC or WG input, so sigma = 4. At the subsequent processing interfaces the open circle shows the internal signal level for coherent WG input and the open square shows the internal signal level for incoherent ADC input. The closed squares show the lower range of the fixed point representation of the signals. The design effort is to ensure that the input system noise remains at the lower bits of the fixed point representation. The reported values for the SST and BST correspond to the input sigma level of 2 bit.

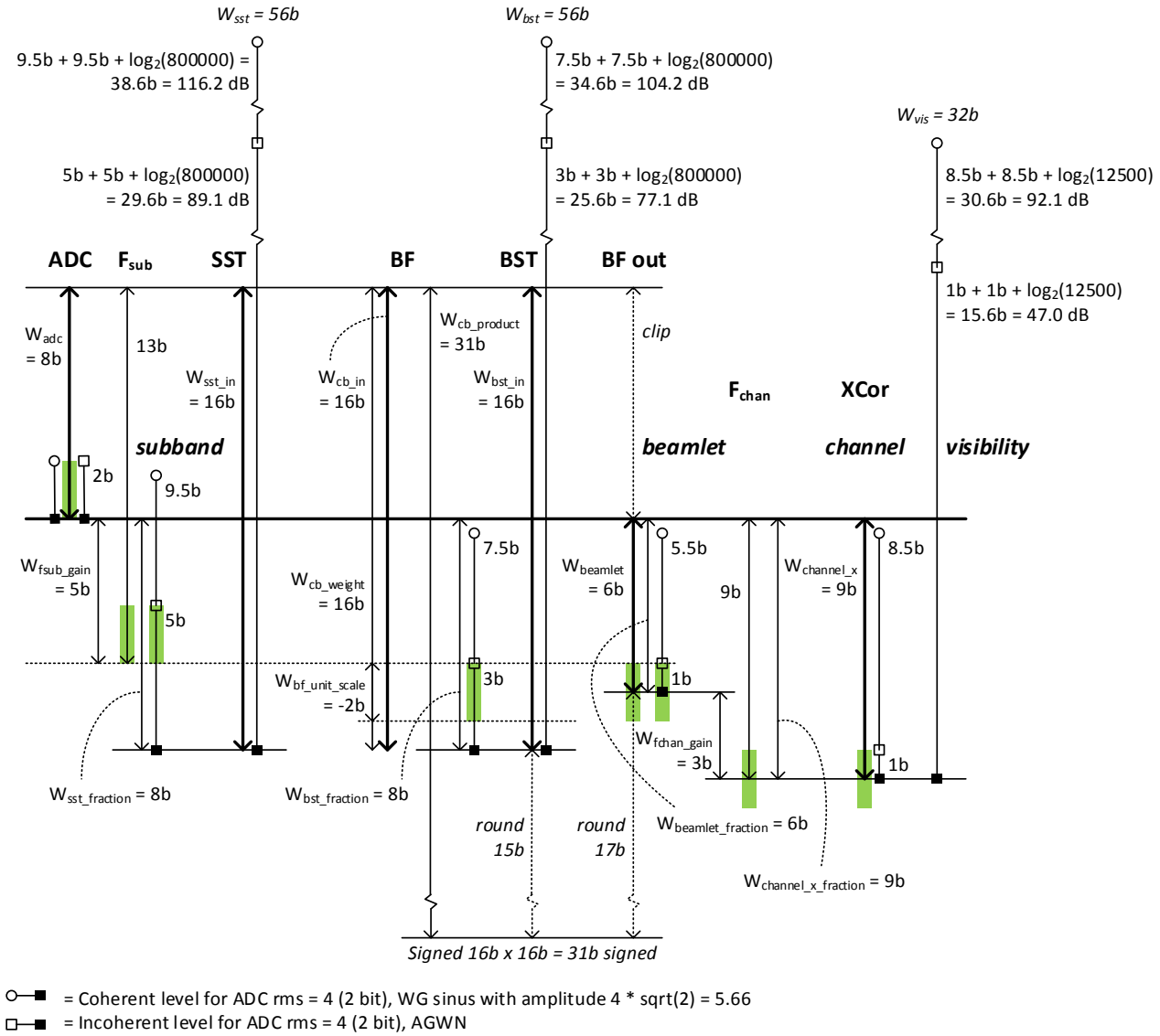

Figure 5: Quantization levels in Apertif BF and X

3.3 Arts

3.3.1 Voltage TAB

The voltage TAB beamformer sums up data from up to N_{dish} = 12 dishes in the WSRT array. All inputs are weighted with same magnitude, but with linear phase to point the TAB [2]. For coherent input the output power will increase by a factor N_{dish} and for incoherent input the output will increase by a factor sqrt(N_{dish}). Therefore the dynamic range of the TAB output increases and the TAB output needs $\log_2(N_{dish} / \text{sqrt}(N_{dish}))$ is $\log_2(\text{sqrt}(12))$ = 1.8 extra bits (see Figure 6).

3.3.2 Arts SC1 fixed point voltage TAB

The voltage TAB in Arts SC1 sums beamlets. For Arts SC1 the TAB weights are $W_{tab\_weight\_sc1}$ = 16 bit, but effectively only 1 bit is used because *sc1_use_qua* = False in the arts_unb1_sc1 implementation [5], so the TAB weights are fixed at 1, so effectively $W_{tab\_unit\_scale\_sc1}$ = -W_tab_max_weight_sc1 = -15. This implies that

the TAB ouput increases with nofDishes for WG input and with sqrt(nofDishes) for noise input. The W_{volt} = 8 bit, so the Arts SC1 has sufficient bits extra bits to fit $\log_2(\text{sqrt}(12))$ = 1.8 bits for nofDishes = 12.

Figure 6 shows the quantization levels within the Apertif BF and the Arts SC1 relative to the input ADC level. The green bar at the input indicates a sigma level of 2 bits for ADC or WG input, so sigma = 4. With one dish the TAB output levels are 2**1 = 2 for incoherent noise input and 2**5.5 = 45.3 for coherent WG input.

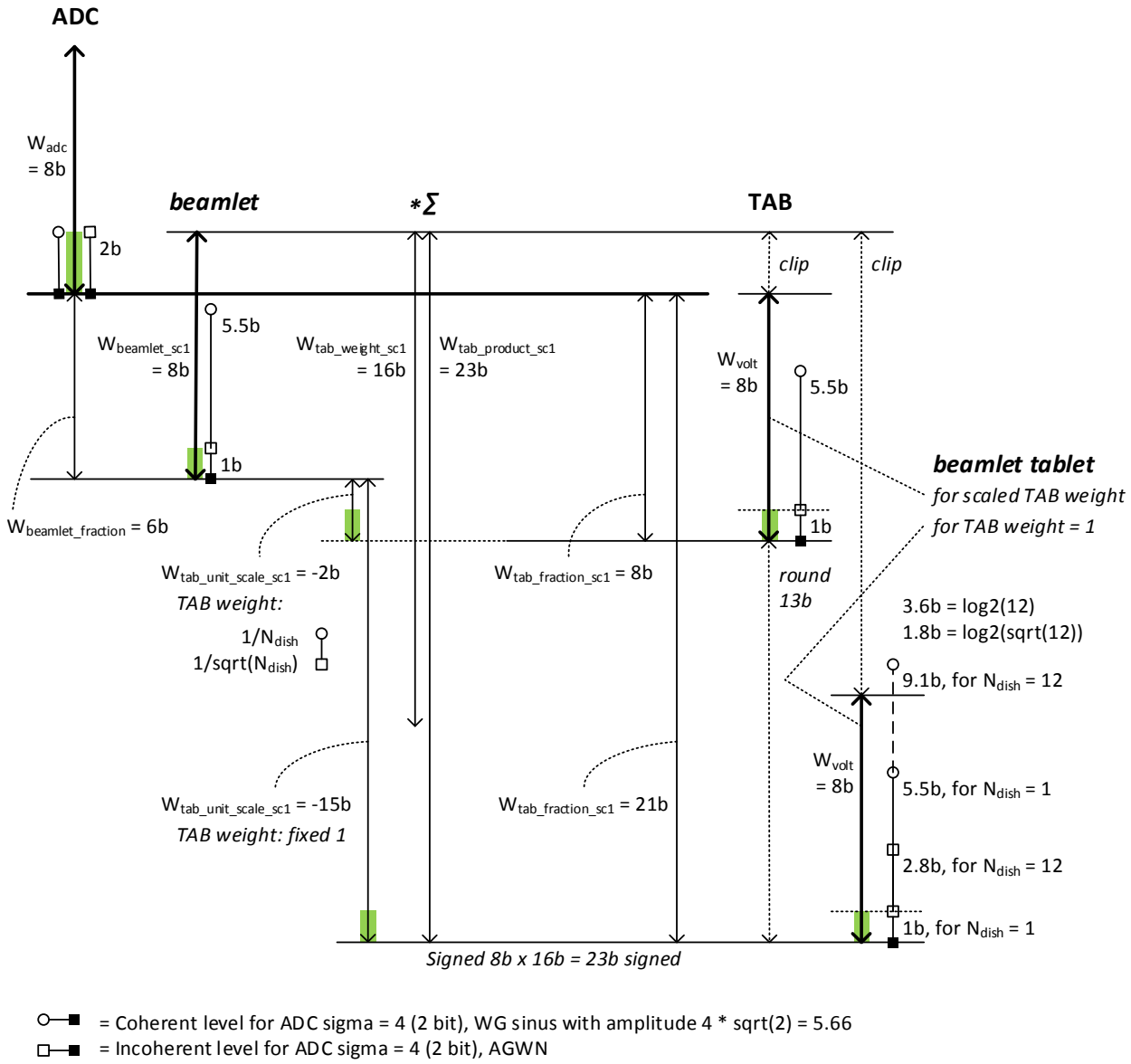

Figure 6: Quantization levels in Apertif BF and Arts SC1 voltage TAB

3.3.3 Stokes I power TAB and power IAB

The voltage TAB in Arts SC4 sums fine channels. The TAB weights are $W_{tab\_weight\_sc4}$ = 9 bits for Arts SC4. The unit gain of the TAB beamformer weights is $W_{tab\_unit\_scale\_sc4}$ = -1 (????). The firmware is coded such that for a TAB weight of 2**($W_{tab\_weight\_sc4}$-1 $W_{tab\_unit\_scale\_sc4}$) - 1 = 127 and only one input the TAB output has the same range as the input, so unit gain. The TAB weights are programmable, but constant during an observation. Dependent on the number of dishes that participate in the observation the TAB weights need to

be scaled by a factor 1/nofDishes for WG input or by a factor 1/sqrt(nofDishes) for noise input, to ensure that the input signal will still fit in the TAB output range.

The IAB beamformer first takes the input powers per fine channel. Then the input powers are integrated over M_{int_ax} = 16 fine channels to get the input powers per Arts BF channel, so with N_{chan_x} = 64 this yields N_{chan_bf} = 4 for Arts. The IAB then sums the input powers from *nofDishes* per Arts BF channel. Hence in total the IAB sums M_{int_ax} * *nofDishes* fine channel input powers. For a single fine channel the mean input power is sigma**2 and the 3 sigma maximum input power is (3*sigma)**2 = 9 sigma**2, so 9 times the mean input power.

The Stokes integration over M_{int_ax} = 16 Arts BF channels in time is done instead over M_{int_ax} = 16 fine channels in frequency. For the output power TAB the fine channel TAB power is integrated. For the output IAB the fine channel power is integrated. With M_{int_ax} = 1 the std(power) is equal to the mean(power). The Stokes power integration of the output power TAB reduces the std(power) of the output power TAB by a factor 1/sqrt(M_int_ax). The std(power) of the output power TAB is independent of *nofDishes*. The std(power) of the output IAB reduces by a factor 1/sqrt(*nofDishes*). Hence in total the std(power) of the output IAB reduces by a factor 1/sqrt(M_{int_ax} * *nofDishes*). This integration implies that the dynamic range of the Stokes values for system noise is reduced and that the output will typically 'never' clip if there is no RFI.

3.3.4 Apertif BF and Arts SC3,4 fixed point power TAB

Figure 7 shows the quantization levels within the Apertif BF and the Arts SC3,4 Stokes I power TAB relative to the input ADC level. The green bar at the input indicates a sigma level of 2 bits for ADC or WG input, so sigma = 4. The TAB weights are scaled by the *nofDishes*, so the Stokes I output level is independent of *nofDishes*. For incoherent noise input the Stokes I output power level is 2**2 = 4.

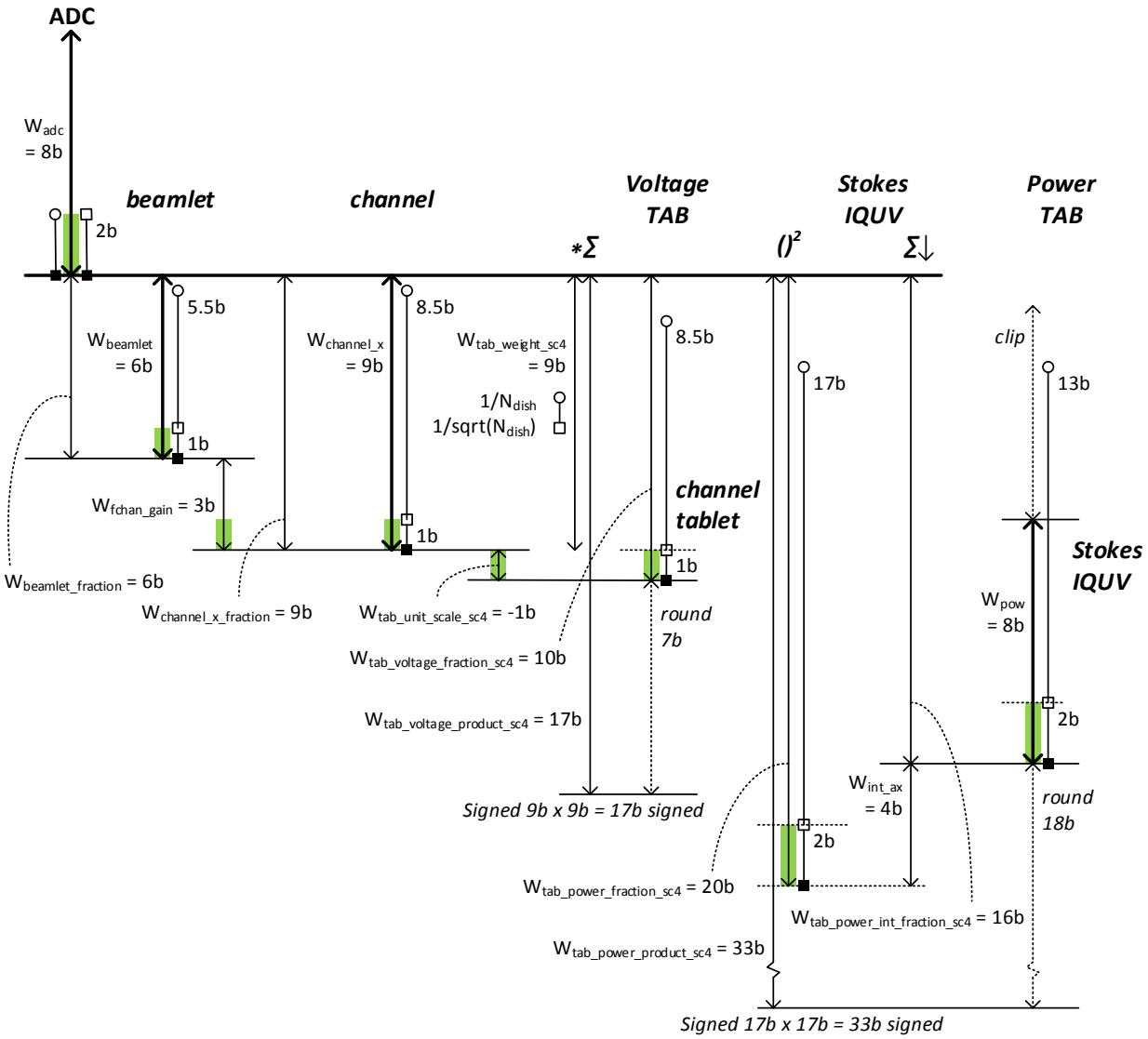

Figure 7: Quantization levels in Arts SC3 and SC4 power TAB

3.3.5 Apertif BF and Arts SC3,4 fixed point power IAB

Figure 8 shows the quantization levels within the Apertif BF and the Arts SC3,4 Stokes I power IAB relative to the input ADC level. The green bar at the input indicates a sigma level of 2 bits for ADC or WG input, so sigma = 4. The IAB gains are scaled by the nofDishes, so the Stokes I output level is independent of nofDishes. For incoherent noise input the Stokes I output power level is 2**2 = 4.

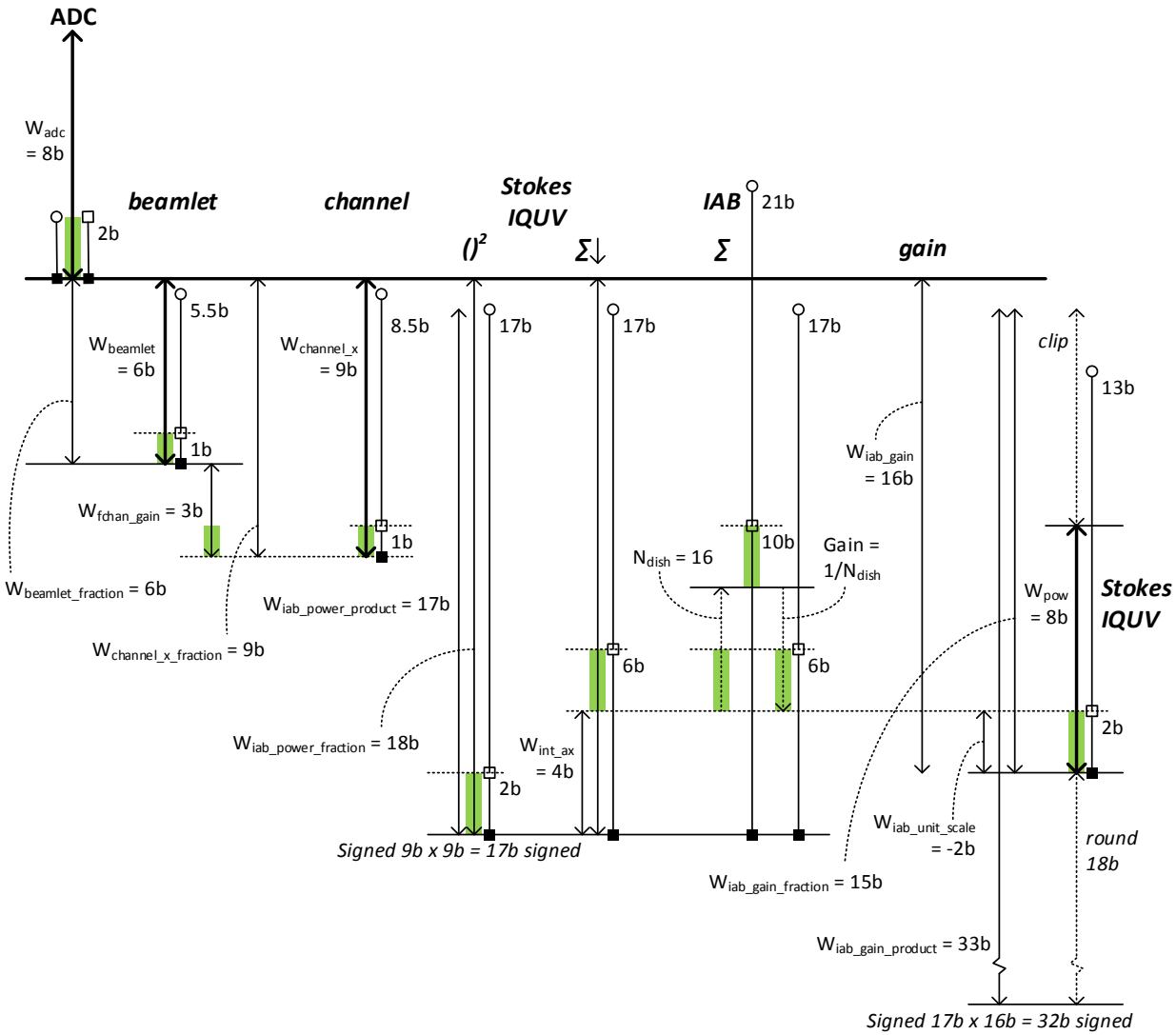

Figure 8: Quantization levels in Arts SC3 and SC4 power IAB

4 Model simulation results

4.1 User interface command line options

Table 1 shows the user interface for the model:

```
> python apertif_arts_firmware model –h
```

Argument	Default	Description
-h		Show this help message and exit
--app	all	Application to model: dish, apertif, arts_sc1, arts_sc4 or all
--model	all	Model: static, dynamic or all
--sky_sigma	4.0	Signal sigma level at ADC input
--wg_enable	False	Model coherent ADC input using WG sinus or model incoherent ADC input using Gaussian white noise
--wg_ampl	$4.0 * 2^{0.5}$	WG amplitude at ADC input
--wg_sweep	0.0	Number of subbands fsub to sweep with WG in dynamic model
--nof_periods	250	Number of channel periods in dynamic model
--nof_dishes	1	Number of dishes
--quantize	False	Use fixed point rounding or no rounding of weights, gains and internal signals
--useplot	False	Use plots or skip plotting
--cb_weight_adjust	1.0	Factor to adjust unit CB weight
--tab_weight_adjust_sc1	1.0	Factor to adjust unit TAB weight for SC1
--tab_weight_adjust_sc4	1.0	Factor to adjust unit TAB weight for SC4
--iab_gain_adjust	1.0	Factor to adjust unit IAB gain for SC4

Table 1: User interface command line arguments

The two --models are:

- A static model to determine the expected internal signal levels as function of the WG amplitude or the ADC noise sigma level.
- A dynamic model using FFT, BF and ()**2 to model the data path signal processing for a WG input signal or a system noise input signal. The FIR filter part of the F_{sub} and F_{chan_x} is not modelled, because the assumption is that they have unit gain. The WG input for test purposes to verify the linearity and dynamic range as function of the WG amplitude and the number of dishes. Noise input for operational purposes to verify that the system noise maintains represented by sufficient LSbits throughout the data path processing chain from ADC via beamlet and fine channel to correlator visibility, Arts SC beamlet TAB, Arts SC4 power TAB and IAB.

In addition within the dynamic model the simulation can also perform a WG frequency sweep by using --wg_sweep > 0 or < 0. In WG frequency sweep mode the reported signal levels are not relevant.

4.2 Logging

The log is obtained using the default arguments:

```
> python apertif_arts_firmware model.py


----------------------------------------------------------------------
-- User settings
----------------------------------------------------------------------
ADC noise input:
. skySigma              = 4.000000
. quantize              = False
. nofDishes             = 1   (= 0.0 bit)

Dynamic model:
. selSub                = 65
. selChanx              = 33
. selChanxFraction      = 0
. nofTchanx             = 250
. nofTsub               = 16000
. nofTs                 = 16384000

----------------------------------------------------------------------
-- Design settings
----------------------------------------------------------------------
Apertif BF:
. N_fft                 = 1024
. N_sub                 = 512
. N_int_sub             = 800000

Apertif XC:
. N_chan_x              = 64
. N_chan_bf             = 4
. N_int_chan_x          = 12500
. cbWeightAdjust        = 1.000000
. cbMaxWeight           = 32767
. cbUnitWeight          = 8192
. cbWeight              = 8192
. cbGain                = 0.250008

Arts SC1:
. sc1_use_qua           = False
. tabWeightAdjustSc1    = 1.000000
. tabUnitWeightSc1      = 1
. tabWeightSc1          = 1
. tabGainSc1            = 0.00003052

Arts SC4:
. tabWeightAdjustSc4    = 1.000000
. iabGainAdjust         = 1.000000
. iabMaxGain            = 32767
. iabUnitGain           = 8192
. iabGainReg            = 8192
. iabGain               = 0.25000763
```

```
-------------------------------------------------------------------
-- Dynamic model settings
-------------------------------------------------------------------
Design:
. fs                          = 800000000.000000 [Hz]
. Ts                          = 1.25 [ns]
. fsub                        = 781250.000000 [Hz]
. Tsub                        = 1.28 [us]
. W_fsub_gain                 = 5 [bit]
. fchan_x                     = 12207.031250 [Hz]
. Tchanx                      = 81.92 [us]
. W_fchan_x_gain              = 3 [bit]
. Tstokesx                    = 81.92 [us]

Dish:
. W_adc                       = 8 [bit]
. W_adc_max                   = 7 [bit], adc max = 127
. W_sst_in                    = 16 [bit]
. W_sst_fraction              = 8 [bit]
. W_cb_weight                 = 16 [bit]
. W_cb_unit_scale             = -2 [bit]
. W_cb_weight_fraction        = 13 [bit]
. W_cb_in                     = 16 [bit]
. W_cb_product                = 31 [bit]
. W_bst_in                    = 16 [bit]
. W_bst_fraction              = 8 [bit]
. W_beamlet_sc1               = 8 [bit]
. W_beamlet                   = 8 [bit]
. W_beamlet_fraction          = 6 [bit]

Apertif XC:
. W_channel_x                 = 9 [bit]
. W_channel_x_fraction        = 9 [bit]

Arts SC1 beamlet TAB:
. W_tab_weight_sc1            = 16 [bit]
. W_tab_unit_scale_sc1        = -15 [bit]
. W_tab_weight_fraction_sc1   = 0 [bit]
. W_beamlet_sc1               = 8 [bit]
. W_tab_product_sc1           = 23 [bit]
. W_tab_fraction_sc1          = 21 [bit]
```

```
Arts SC4 TAB:
. W_channel_x                  = 9 [bit]
. W_channel_x_fraction         = 9 [bit]
. W_tab_weight_sc4             = 9 [bit]
. W_tab_unit_scale_sc4         = -1 [bit]
. W_tab_weight_fraction_sc4    = 7 [bit]
. tabUnitWeightSc4             = 128
. tabWeightSc4                 = 128
. tabGainSc4                   = 0.50196078
. W_beamlet                    = 8 [bit]
. W_tab_voltage_product_sc4    = 17 [bit]
. W_tab_voltage_fraction_sc4   = 10 [bit]
. W_tab_power_product_sc4      = 33 [bit]
. W_tab_power_fraction_sc4     = 20 [bit]
. W_tab_power_int_fraction_sc4 = 16 [bit]
. W_int_ax                     = 4 [bit]
. M_int_ax                     = 16

Arts SC4 IAB:
. W_channel_x                  = 9 [bit]
. W_channel_x_fraction         = 9 [bit]
. W_int_ax                     = 4 [bit]
. M_int_ax                     = 16
. W_iab_gain                   = 16 [bit]
. W_iab_unit_scale             = -2 [bit]
. W_iab_gain_fraction          = 13 [bit]
. W_iab_power_product          = 17 [bit]
. W_iab_power_fraction         = 18 [bit]
. W_iab_gain_product           = 32 [bit]
. W_iab_gain_fraction          = 16 [bit]
```

```
-----------------------------------------------------------------------
-- Dynamic model results
-----------------------------------------------------------------------
Apertif BF internal levels (excluding sign bit) for incoherent ADC noise input:
. ADC input rms                =                 4.00              (=  2.0 [bit])
. SST input rms complex        =                31.95              (=  5.0 [bit])
. SST input rms                =                22.59              (=  4.5 [bit])
. BST input rms complex        =                 7.99              (=  3.0 [bit])
. BST input rms                =                 5.65              (=  2.5 [bit])
. CB beamlet rms complex       =                 2.00              (=  1.0 [bit])
. CB beamlet rms               =                 1.41              (=  0.5 [bit])
. SST power                    =         816459834.75  =   89.12 dB (= 29.6 [bit])
. BST power                    =          51031854.36  =   77.08 dB (= 25.6 [bit])

Apertif XC output for incoherent ADC noise input:
. Fine channel rms complex     =                 2.00              (=  1.0 [bit])
. Fine channel rms             =                 1.42              (=  0.5 [bit])
. XC auto power                =             50248.60  =   47.01 dB (= 15.6 [bit])

Arts SC1 beamlet TAB internal levels (excluding sign bit) for incoherent ADC noise input:
. Output TAB rms complex       =                 2.00              (=  1.0 [bit])
. Output TAB rms               =                 1.41              (=  0.5 [bit])

Arts SC4 TAB internal levels (excluding sign bit) for incoherent ADC noise input:
. Fine channel rms complex     =                 2.00              (=  1.0 [bit])
. Fine channel rms             =                 1.42              (=  0.5 [bit])
. Fine channel TAB rms complex =                 2.01              (=  1.0 [bit])
. Fine channel TAB rms         =                 1.42              (=  0.5 [bit])
. Fine channel TAB power       =                 4.05              (=  2.0 [bit])
. Fine channel TAB power std   =                 3.70              (=  1.9 [bit])
. Output TAB power             =                 4.05              (=  2.0 [bit])
. Output TAB power std         =                 0.93              (= -0.1 [bit])

Arts SC4 IAB internal levels (excluding sign bit) for incoherent ADC noise input:
. Fine channel rms complex     =                 2.00              (=  1.0 [bit])
. Fine channel rms             =                 1.42              (=  0.5 [bit])
. Fine channel power           =                 4.02              (=  2.0 [bit])
. Integrated channel power     =                64.32              (=  6.0 [bit])
. Integrated IAB power         =                64.32              (=  6.0 [bit])
. Output IAB power             =                 4.02              (=  2.0 [bit])
. Output IAB power std         =                 0.92              (= -0.1 [bit])
```

The reported levels, that are marked in **bold** in the log above, agree with the levels shown in the fixed point quantization level figures. For Apertif BF and XC:

- ADC input rms = 4 (= 2 bit) as in Figure 5
- SST power = 89.12 dB as in Figure 5
- BST power = 77.08 dB as in Figure 5
- CB beamlet rms complex = 2.0 (= 1.0 bit) as in Figure 5
- Fine channel rms complex is 2.0 (= 1.0 bit) as in Figure 5
- XC auto power = 47.01 dB as in Figure 5

For Arts SC1:

- Output TAB rms complex = 2.0 (= 1.0 bit) as in Figure 6

For Arts SC4:

- Output TAB power = 4.05 (= 2.0 bit) as in Figure 7
- Output IAB power = 4.02 (= 2.0 bit) as in Figure 8

4.3 Plots

4.3.1 Static and dynamic model

The plots in the figures below are obtained using the default arguments and --useplot:

```
> python apertif_arts_firmware model.py --useplot
```

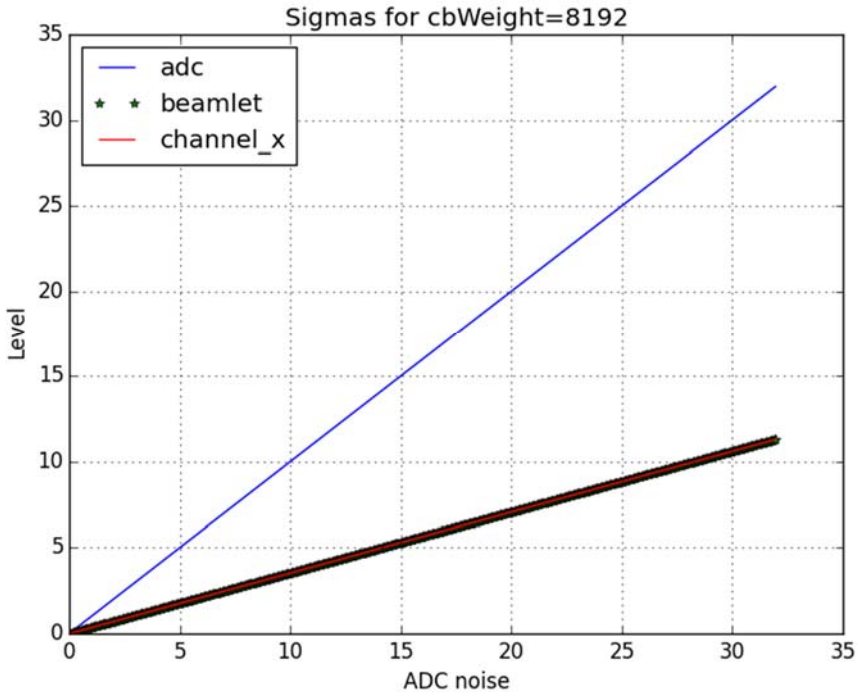

Figure 9: apertif_arts_firmware_model_static_noise_sigmas

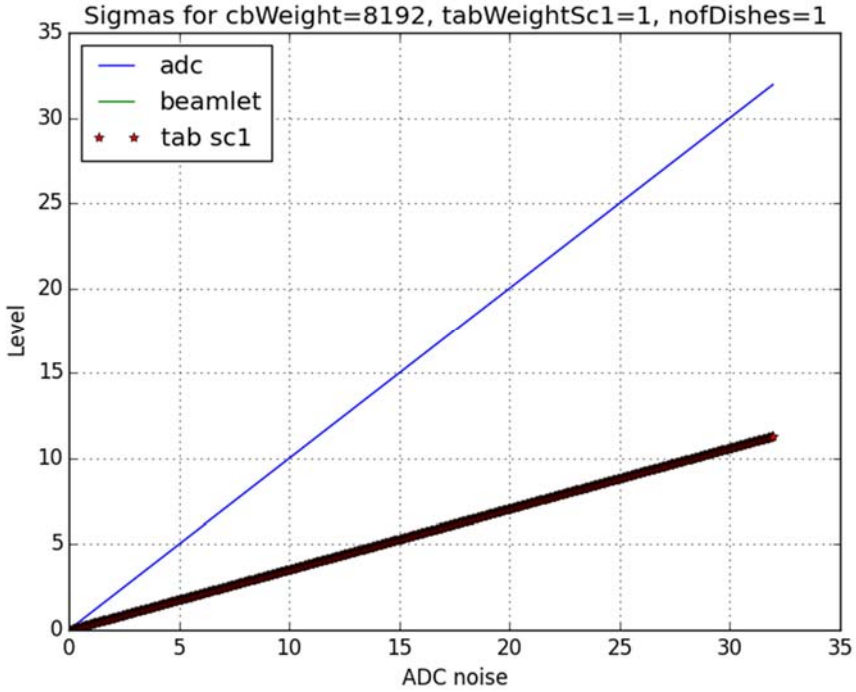

Figure 10: apertif_arts_firmware_model_static_noise_sigmas_sc1

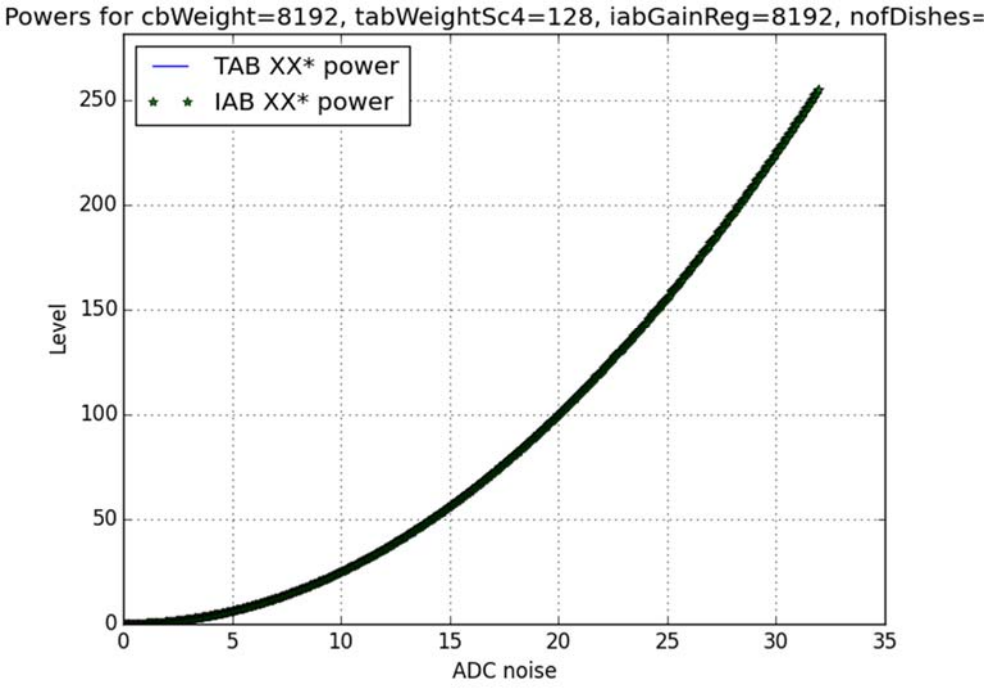

Figure 11: apertif_arts_firmware_model_static_noise_powers_sc4

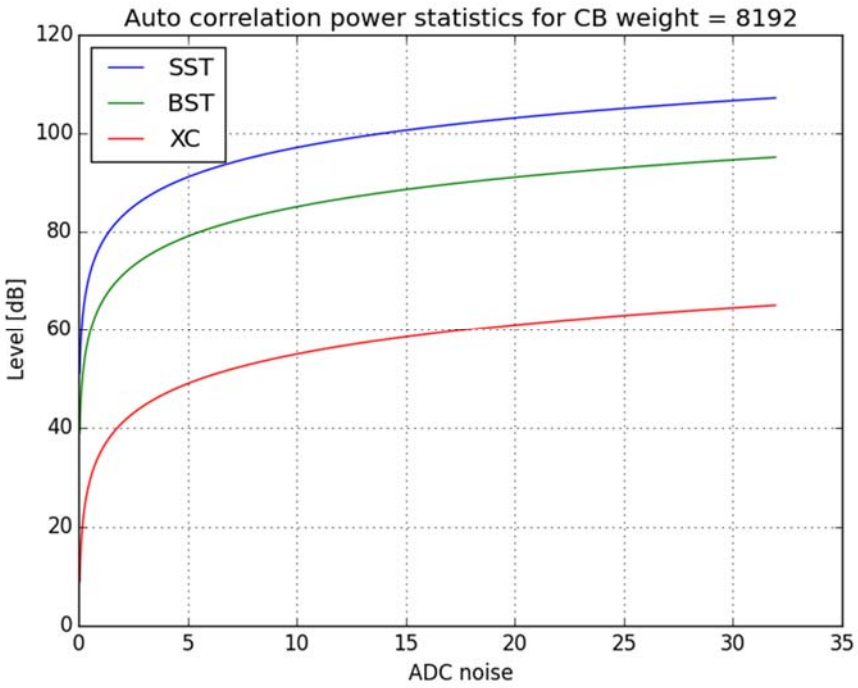

Figure 12: apertif_arts_firmware_model_static_noise_auto_powers

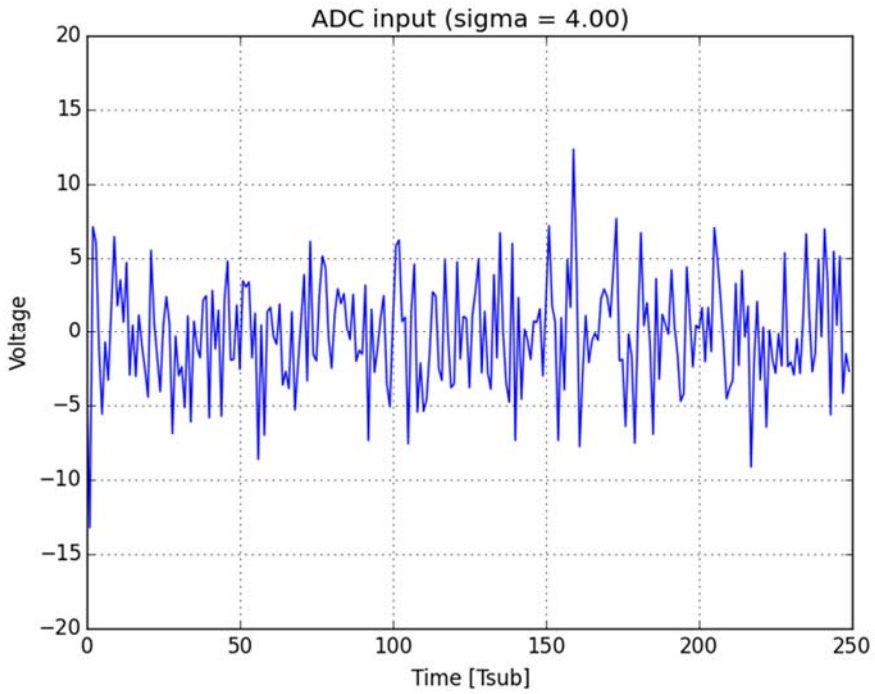

Figure 13: apertif_arts_firmware_model_dynamic_adc

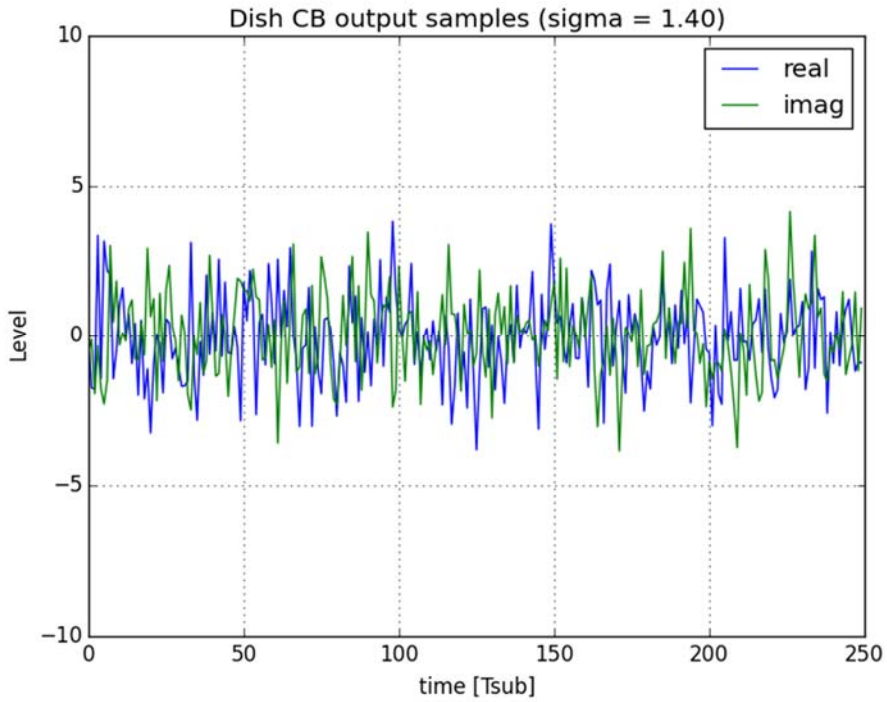

Figure 14: apertif_arts_firmware_model_dynamic_beamlet

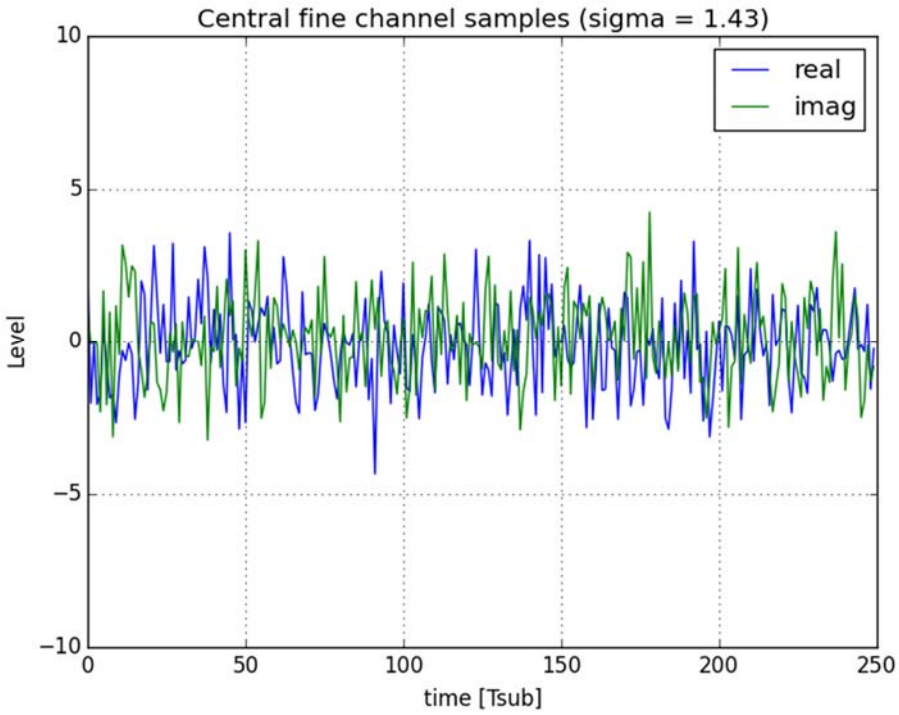

Figure 15: apertif_arts_firmware_model_dynamic_fine_channel

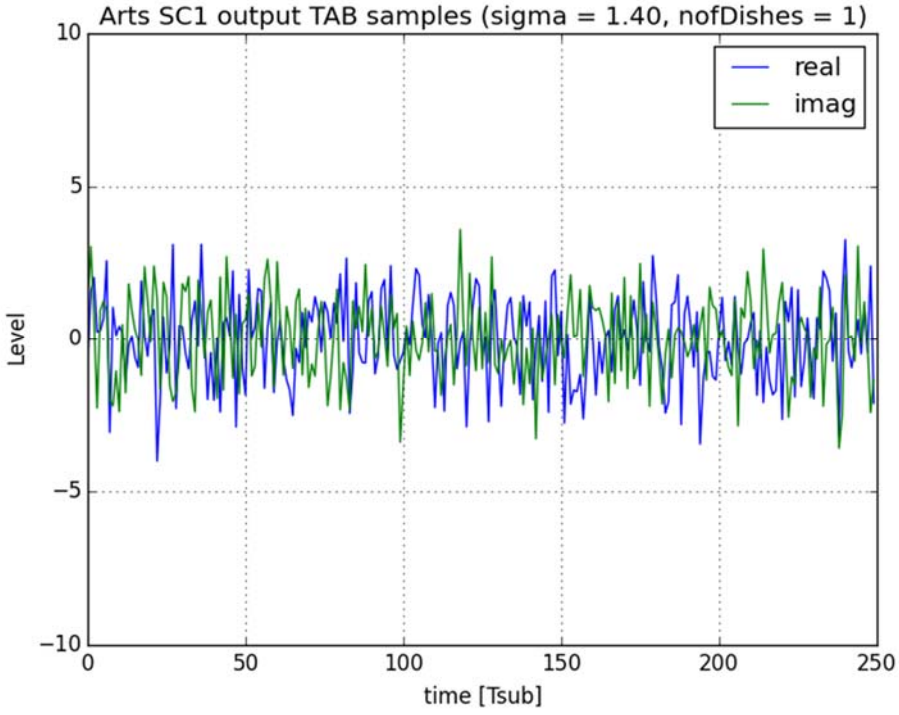

Figure 16: apertif_arts_firmware_model_dynamic_beamlet_tab

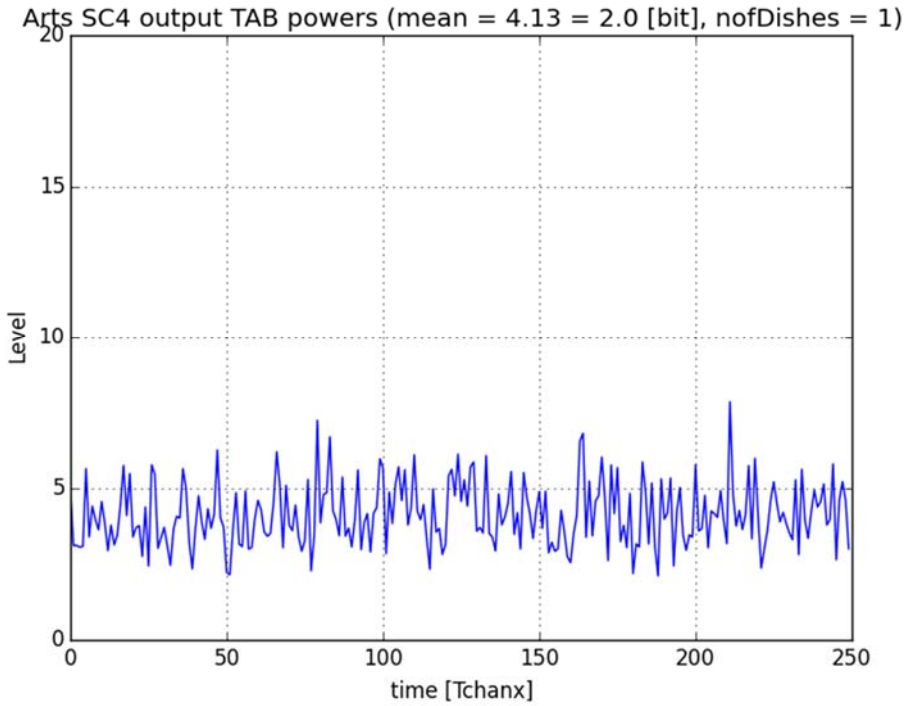

Figure 17: apertif_arts_firmware_model_dynamic_power_tab

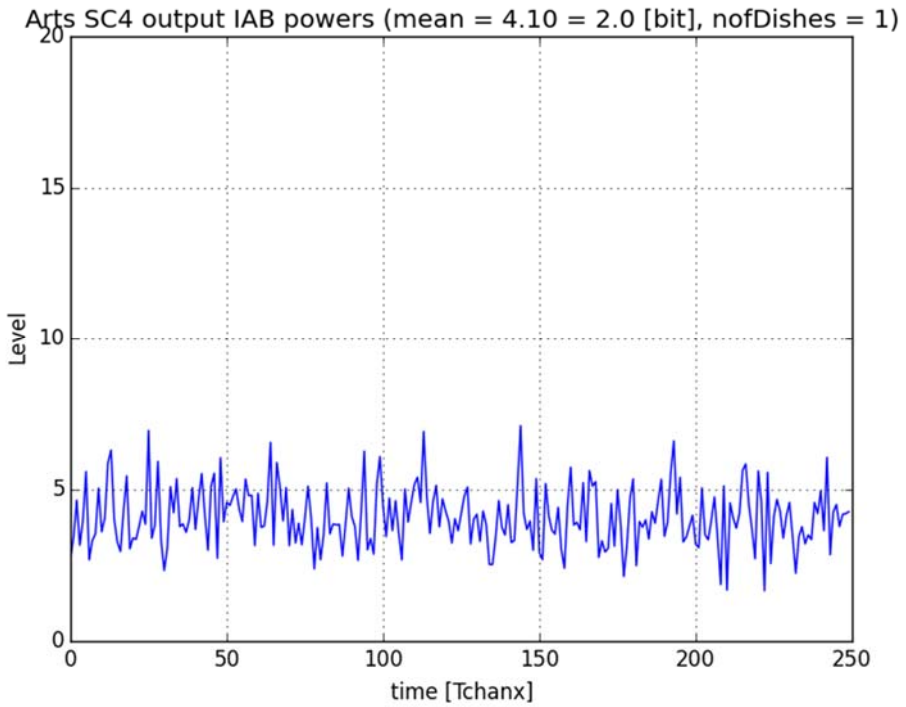

Figure 18: apertif_arts_firmware_model_dynamic_power_iab

4.3.2 Sweep WG model

The plots in the figures below are obtained using the default arguments and wg_sweep = 5 to frequency sweep over 5 subbands:

```
> python apertif_arts_firmware model.py --useplot --model dynamic --wg_sweep 5.0
```

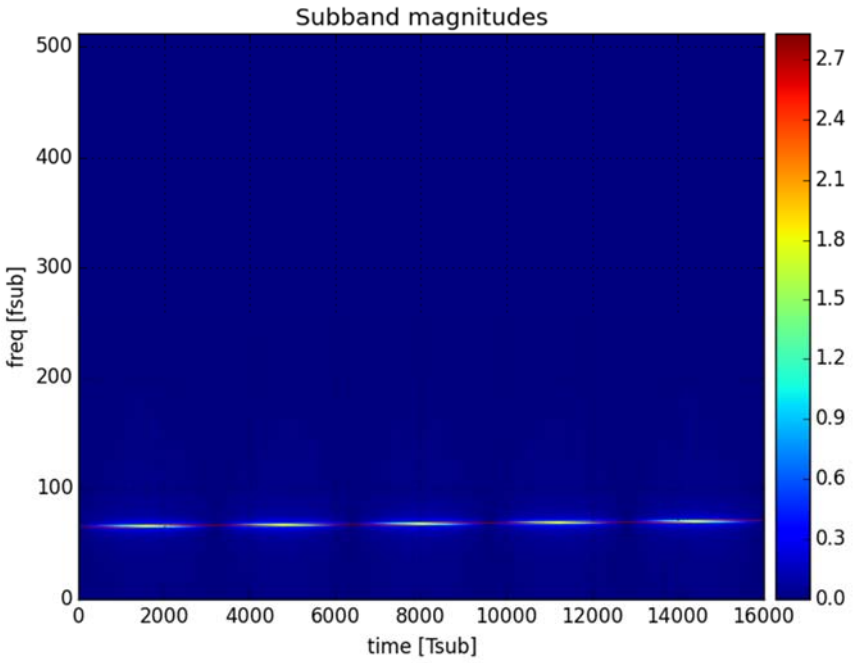

Figure 19: apertif_arts_firmware_model_wg_sweep_subband_spectrum

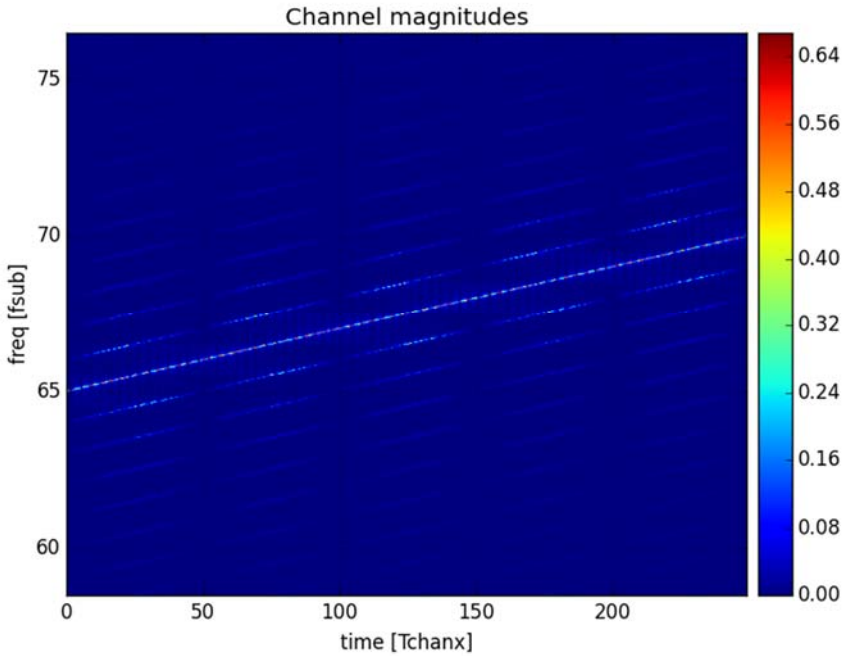

Figure 20: apertif_arts_firmware_model_wg_sweep_fine_channel_spectrum

5 Hardware verification results

The ADC input level can be controlled with the attenuator on ADU, e.g. using *util_adu_i2c_commander.py* [7]. The BF weights unit value of 8192 matches the preferred setting. In operations all are controlled by the MAC software and monitored by WinCC (Artamis). The commands in this section use the DESP Python commands [7] and are only shown as example.

The following commands from *pi_apertif_system.py* [6] can be used on *ccu-corr* to log and plot internal signals. The PAF element at unb 3, bn 2, sp 0 is the global X pol SP-56 at the center of the PAF, the Y pol SP-56 is on unb 7. For more usage examples see [7].

5.1 ADUH: ADC samples

```
> ssh lcu-rt7 -X "python $UPE/peripherals/util_aduh_monitor.py --unb 3,7 --bn 2 --sp 0 -n
5 -v 5 --useplot"
```

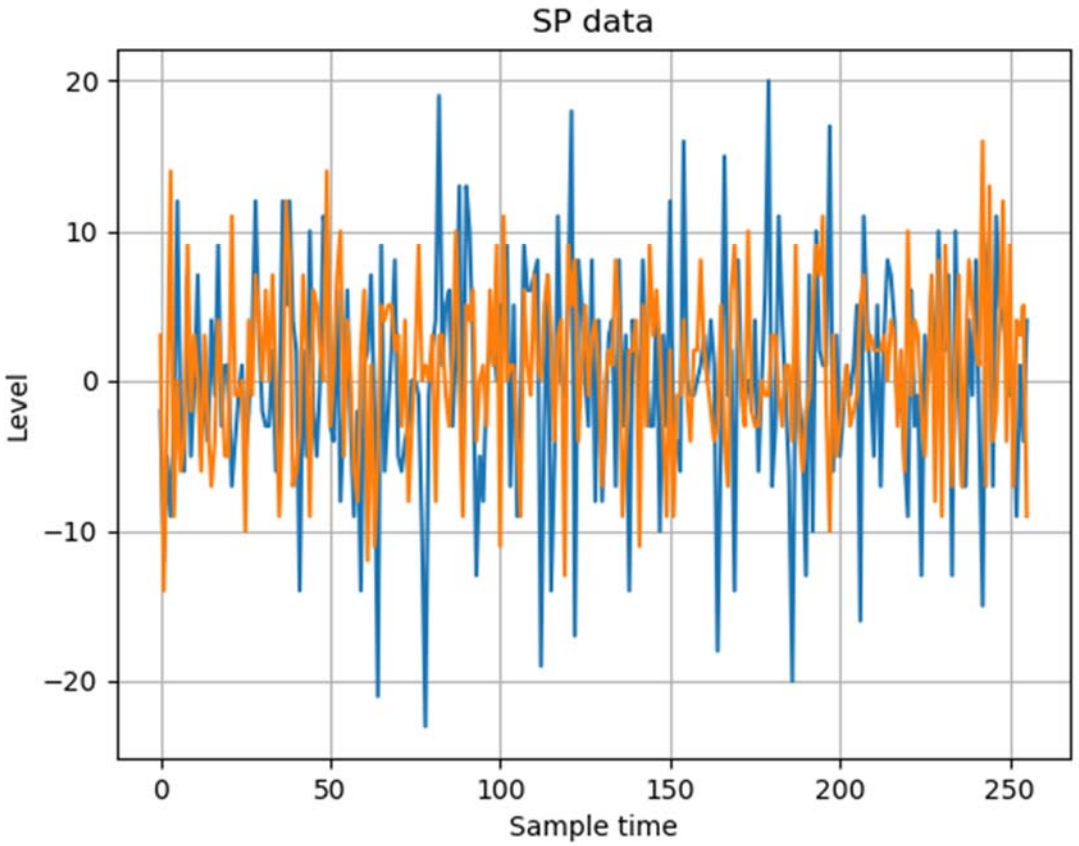

Figure 21: Captured ADC samples

The reported statistics are:

```
[2019:02:20 08:57:10] - (5) UTIL_ADUH_MON - AMON - UNB-3, BN-2, I-0: Read SP power =
52.91, SP sigma =    7.27     (equivalent WG ampl =   10.29)
[2019:02:20 08:57:10] - (5) UTIL_ADUH_MON - AMON - UNB-7, BN-2, I-0: Read SP power =
37.87, SP sigma =    6.15     (equivalent WG ampl =    8.70)
```

5.2 SST: subband statistics

```
> ssh lcu-rt7 -X "python $UPE/peripherals/pi_apertif_system.py --unb 3,7 --bn 2 --sp 0 --
cmd 10 -v 5 --useplot"
```

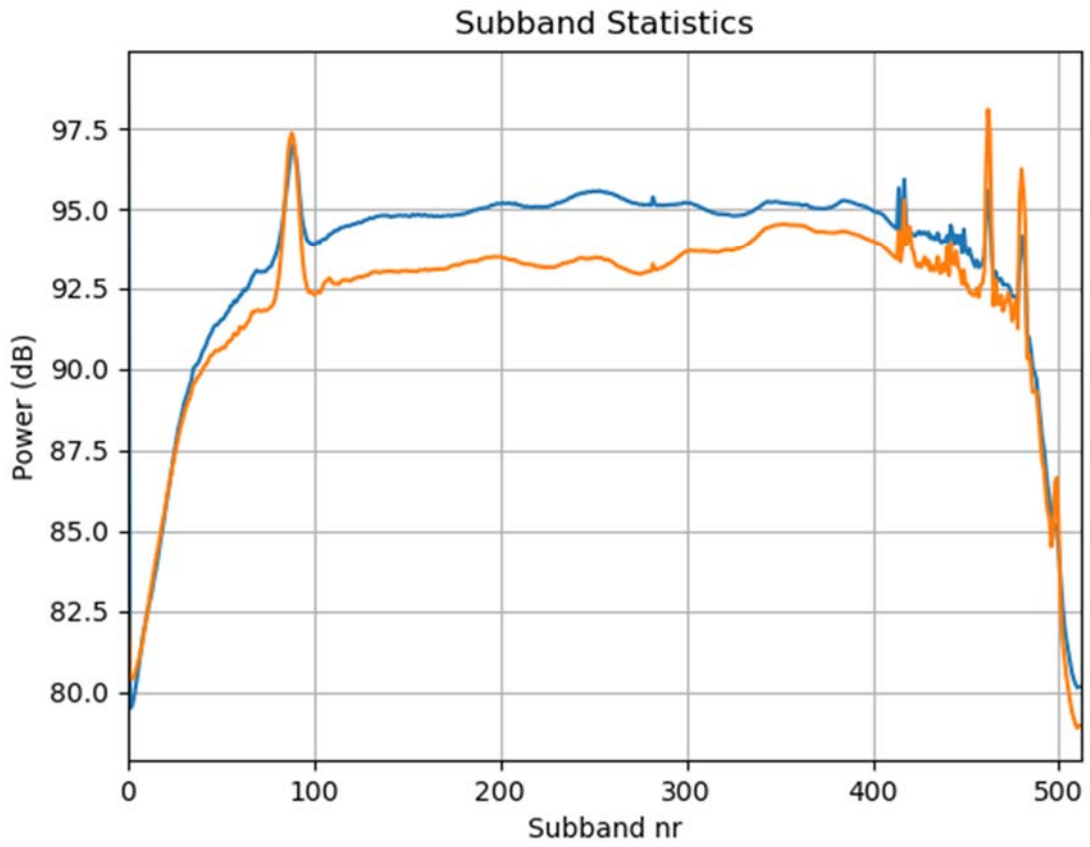

Figure 22: Subband statistics

The --cmd 10 reports the log_pol_subband_sst_data_max value and subband index which is useful when the WG is used, because then the signal level in one subband is dominant. For ADC noise input it is more useful to know the sum of the subband powers and that is reported with --cmd 11. The reported statistics with --cmd 11 were read a few seconds earlier but show similar SP sigma values as with the ADUH in section 5.2:

```
[2019:02:20 08:56:54] - (3) PI_APR - cmd-11: SST - log_sp_subband_sst_data_sum: pol 0, SP
56, SST sum =   1341234073778, SP sigma =    7.153, SP ampl =   10.120
[2019:02:20 08:56:54] - (3) PI_APR - cmd-11: SST - log_sp_subband_sst_data_sum: pol 1, SP
56, SST sum =    966655008779, SP sigma =    6.072, SP ampl =    8.590
```

The SST sum 1341234073778 is the sum of N_{sub} = 512 subband powers, so on average the subband power level is 10 $\log_{10}$(1341234073778 / 512) = 94.2 dB, which matches the upper blue curve in Figure 22. Similar 10 $\log_{10}$(966655008779/ 512) = 92.8 dB average subband power, which matches the lower orange curve in Figure 22.

5.3 Beam data

The following commands require that he BF weights for SP-56 have been set to unit default 8192, or at least not 0. For more elaborate results see [4].

5.3.1 BST: beamlet statistics

Log and plot beamlet statistics (requires that the BF weights have been set):

```
> ssh lcu-rt7 -X "python $UPE/peripherals/pi_apertif_system.py --unb 0:7 --fn 0:3 --
cbeams 0 --pol 0,1 --cmd 71 -v 5 --useplot"
```

5.3.2 XCor DB mesh: beamlet data

Log and plot beamlets time series data (requires that the BF weights have been set):

```
> python $UPE/peripherals/util_diag_data_buffer.py -v 5 --unb 0 --fn 0:3 --bn 0:3 -s MESH
--ramsize 512 -r 0 -n 10 -w 8 --useplot
```

5.3.3 XCor DB output: visibilities

Log channel visibilitities and aggregated absolute subband visibilities:

```
> python $UPE/peripherals/pi_apertif_system.py --cmd 102 -v 3 --tel b --unb 0 --fn 0:3 --
bn 0:3 --cbeams 0 --sub 64 --chan 32 --ramsize 39936
```