SKA Project Series
Structure and processing of the F section for SKA

G. Comoretto[1]

[1]INAF - Osservatorio Astrofisico di Arcetri

Arcetri Technical Report N° 4/2014
20-dec-2014

Abstract

The three (Low, Mid and Survey) SKA correlators share a similar structure, with a FX architecture. The F part has a total of the order of 2^{18} channels, with a dual stage filter-bank. The first stage provides channelization with overlapping channels, while the second provides non overlapping channels with good channel-to-channel insulation.

The channelization parameters (filter shape vs. filter complexity, overlapping amount, etc) are similar for the correlators, despite important differences in bandwidth, physical implementation, etc. We discuss these parameters, with considerations on the specificity of the three correlators.

The delay and fringe compensation functions are also implemented inside the channelizer, and are discussed here.

Contents

List of Tables

List of Figures

List of acronyms

ADD: Architecture Design Document

ARC: Architecture Work Package

ASIC: Application Specific Integrated Circuit

CAD: Computer Aided Design

COTS: Commercial Off-The-Shelf

CSP: Central Signal Processor

DIF: Decimation In Frequency

DSH: Dish Element or Consortium

DSP: Digital Signal Processing

ECP: Engineering Change Proposal

EICD: External Interface Control Document

EMC: Electromagnetic Compatibility

EMI: Electromagnetic Interface

FFT: Fast Fourier Transformation

FIFO: First In First Out

FPGA: Field Programmable Gate Array

GPU: General Processing Unit

HDL: High Level Design Language

ICD: Interface Control Document

IICD: Internal Interface Control Document

I/O: Input/Output

IP: Intellectual Property

LFAA: Low Frequency Aperture Array Element or Consortium

MATLAB: MATLAB simulation language and application

M&C: Monitor and Control

MOD: Modelling Work Package

PDF: Portable Document Format

PDR: Preliminary Design Review

PHS&T: Packaging, Handling, Storage and Transportation

PIP: Physical Implementation Proposal

PMX: PowerMX

PSS: Pulsar Search

PST: Pulsar Timing

RFI: Radio Frequency Interference

RMP: Risk Management Plan

RMS: Root Mean Square

RR: Risk Register

RS: Requirement Specification

SAD: System Baseline Design

SADT: Signal and Data Transport Element or Consortium

SDE: Software Development Environment (Work Package)

SDP: Science Data Processing

SKA: Square Kilometre Array

SKAO: SKA Organisation (or office)

SKA-SA: SKA South Africa

SNR: Signal to Noise Ratio

SOF: Start Of Frame

SOP: Start Of Packet

SRR: System Requirements Review

SW: Software

SYSML: System Engineering Simulation Language and application

TBC: To be confirmed

TBD: To be decided

TDT: Time Domain Team

TMF: Time multiplexing factor

UML: Unified Modelling Language

VLBI: Very Large Baseline Interferometry

VPL: Verification Planning Work Package

WBS: Work Breakdown Structure

WOLA: Weigth, Overlap and Add operation. Performed in the filter section of a polyphase channelizer

WP: Work Package

1 Introduction

The central signal processor (CSP) for the three SKA telescopes always include a correlator. Each correlator is built around a generic FX architecture. Signal from each antenna (or each antenna station) is processed in an F processor, to obtain a set of finely channelized frequency channels. A subset of channels are sent, for all antennas, to a X processor, that computes the cross correlations for all baselines in that frequency range. In this report the F processor is described.

The F section is also responsible for delay/fringe tracking of each antenna. The cross connect between the F and X section can be quite complex, with a all-to-all topology. Therefore a partial corner turn operation, exchanging antennas and frequency channels, is performed in various stages of the F section, where convenient.

1.1 General specifications

The requirements for the F processors are directly derived from the Level 2 SKA-CSP requirements[3], and are similar for the three telescopes. They are summarized here. Numbers in parenthesis refer to the requirement number, e.g. 2134-02 refers to level 2 specification SKA1-CSP_REQ-2134-02.

- Spectral resolution is close to 2^{18} spectral channels, and is definied as the observed useful bandwidth divided by the final channel spacing. The actual number of channels may be higher, due to discarded portion of the input digitized band and other implementation details (e.g. the oversampling factor) in the signal processing chain. [2148-00, 2195-01 to 2195-05, 2250-01]
- Spectral coverage is continuous over the observed band, i.e. no discontinuities or missing regions are allowed
- Rejection of signals in non adjacent spectral channels must be > 60 dB, especially to avoid RFI contamination. Higher rejection over at least portion of the bandwidth is highly desirable. [2810-00, 2803-00, 2807-00]
- Adjacent channel insulation must be > 60 dB starting at 1/2 channel nominal width from the channel border. The rationale for this is that a monochromatic interfering signal should contaminate at most two adjacent channels, irrespective from its position in the channelization grid. [2149-01, 2196-02]
- Power response across multiple channels, i.e. the sum of the integrated power, should be equal to the integral of the ideal, infinite resolution power spectrum. In particular, the response at the channel edge should be -3 dB (half power on each channel). [2812-00, 2804-00, 2809-00]
- Frequency response, after calibration and correction of the systematic, deterministic effects, should be better than ± 0.3 dB. Frequency response should not introduce significant variations of the noise across the bandwidth, position errors or frequency offsets. [2811-00, 2805-00, 2808-00]
- Frequency broadening of a narrow line due to actual channelizer filter shape should not exceed 10% of the broadening due to an ideal rectangular filter.
- Total SNR degradation, with respect to an ideal, infinite precision, correlator must not exceed 2%. This includes correlation and input signal quantization, leaving a budget of around 0.5% for the channelizer. [2678-00, 2762-00, 2813-00]

2 Signal processing structure

Although it is possible (and is currently often done) to implement FFT channelizers with up to million spectral channels, this is not practical for FPGA based implementations, basically due to the huge amount of memory required in the associated polyphase filter. A two stage architecture has thus been employed, with a single large corner turner memory between the stages.

Both stages use a polyphase filter-bank architecture, with a small (15-20%) overlap for the first stage, and with no overlap for the second stage. A corner turner is required between the two stages, to present a complete sequence of samples for a given coarse frequency channel to the second filter-bank. The delay compensation is

performed applying a phase slope after both the first and the second stage channelization, and introducing a bulk delay between the two stages. Different delay compensation schemes can be used, however, e.g. inserting the bulk delay before any channelization.

The Low and possibly the Survey correlators implement the first stage channelizer before sending the data to the correlator, while the Mid correlator analyses the data directly sampled at the antennas, with both stages in the CSP element. Most of the considerations discussed here, however, are independent from the physical location of the stages.

The three correlators have very different bandwidths, and therefore analyze the signal using time multiplexed structures with a time multiplexing factor (TMF) that could vary between 2 and 32, depending also on the maximum usable processing clock in the FPGA. Apart from implementation details in the first stage channelizer, the proposed structure is however identical.

The SysML description of the complete channelizer structure is shown in fig. 1.

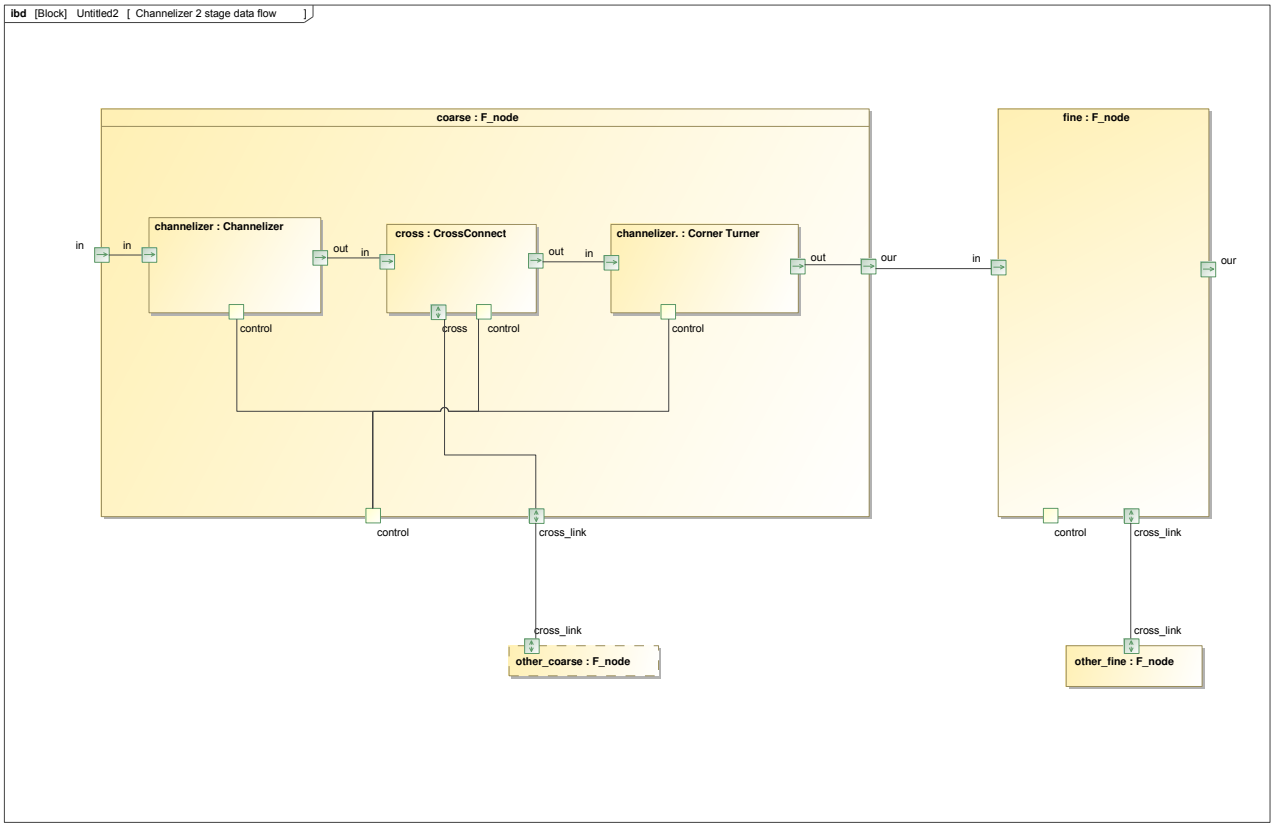


Figure 1: Two stage channelizer structure

The complete channelizer is composed of two similar structures, denoted as *F nodes*, so just one is exploded in the diagram.

Each F node is composed of a channelizer (polyphase filter-bank), a cross connect to provide a portion of the antenna to frequency channel corner turn, and a frequency-time corner turn. The first corner turn exchanges data among adjacent antennas, in order to process in the following stages only a subsection of the frequency channels for multiple antennas (section 3.7). The second corner turn is basically a large memory in which the processed data for an elementary integration time is stored. Data is written in memory as processed from the channelizer, in frames composed of all frequency channels (or a fraction of frequency channels and multiple antennas) for a single time value, and read back one channel at a time for the whole integration period, as required by the next stage channelizer (or X section).

Each channelizer processes the data with the structure shown in fig. 2. The delay subsection (delay model,

coarse and fine delay) may be present either in the coarse or fine F node. The fine delay after the FFT is required only if the correction is performed in the second stage (fine) F node. Delay subsection is described in section 4.

The polyphase filter-bank and the FFT have different structure in the two stages: in the first stage the input is (usually) real, and the polyphase filter implements an oversampling of the input data. The second stage filter-bank always operates on complex valued quantities, and is critically sampled (no oversampling).

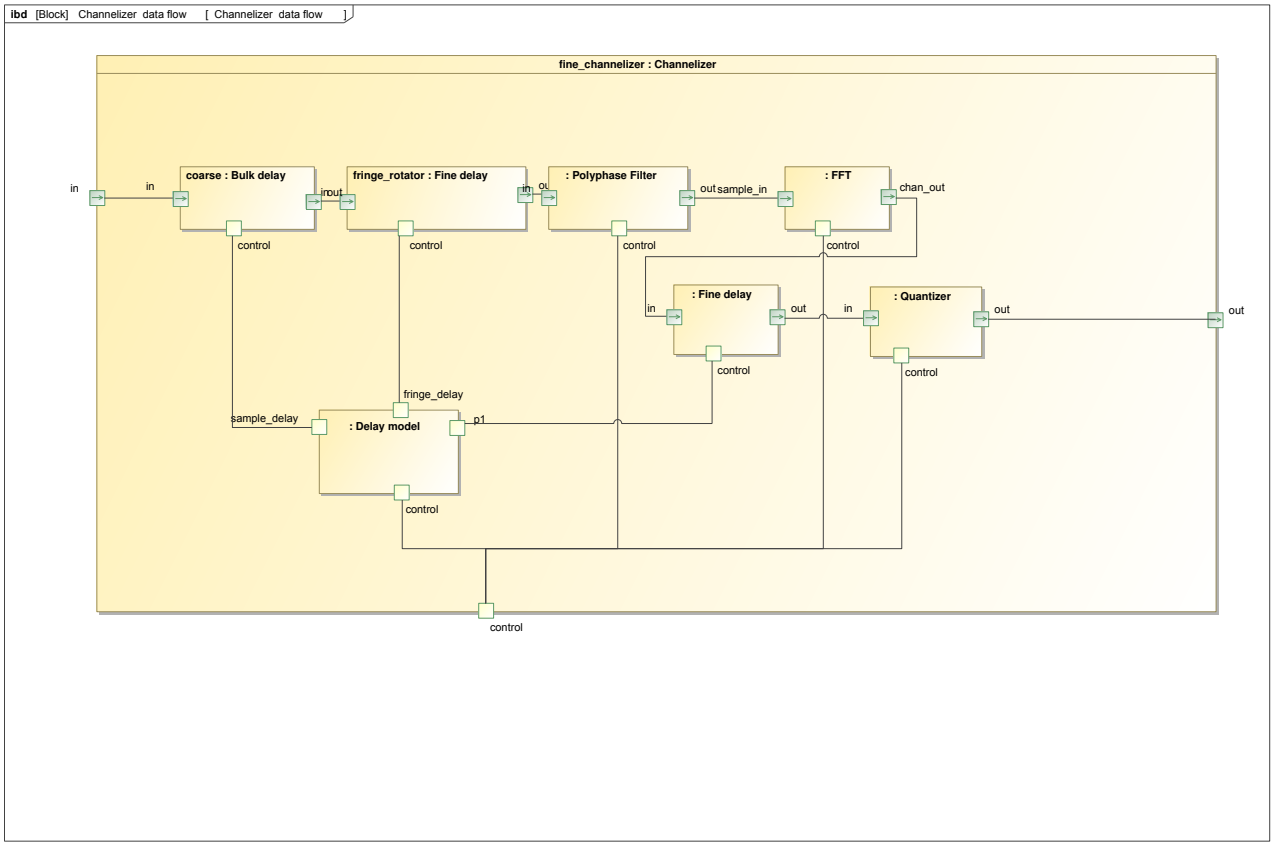


Figure 2: Channelizer data flow

3 First stage (coarse) polyphase filter-bank

The first stage of the channelizer takes real samples, at the ADC sampling frequency, and produces a first set of overlapping, complex valued, frequency channels. In the following sections we will discuss the number of channels implemented in each stage, the amount of overlapping, and the methodology for the filter design.

3.1 Number of frequency channels in the two stages

Final frequency resolution is specified as roughly 250K channels across the correlator bandwidth. The actual number of channels required in the channelizer is however higher, because after each channelization a portion of the spectrum is discarded.

First (coarse) channelized sees all the bandwidth digitized by the ADC, that includes the guard bands of the anti-aliasing analog filter. For typical ADC systems this amounts in a fraction of 20-35% of the digitized band to be discarded. The second channelizer removes the overlapping portions of each channel, that amount

to 14-18%. As a consequence, the total number of channels, depending on the oversampling factor, is between 2^{18} and 2^{19} input sample rate (expressed as ADC bandwidth) and required resolution.

Possible input sample rates for the three telescopes, and requested spectral resolution are listed in table 1, together with the resulting total number of spectral channels and actual resolution. Parameters are taken from level 2 requirements[3]. The actual number of *processed* channels is lower, of the order of 2^{18}, as the discarded channels at each stage are not processed.

Telescope	Input BW	Required resol.	O_f	N. of chans	Actual resol.
LFAA	400 MHz	1 kHz ±20%	1.185	2^{19}	904 Hz
Mid (Dish) band 1	700 MHz	2.7 kHz ±30%	1.143	2^{18}	3.05 kHz
Mid (Dish) band 5	5 GHz	19.5 kHz $+16-20\%$	1.143	2^{18}	21.8 kHz
Survey	500 MHz	1.95 ±30%	1.185	2^{18}	2.26 kHz

Table 1: Examples of needed number of channels depending on input sample rate, requested resolution and oversampling factor O_f

In a 2 stage architecture the total number of channels can be factorized between the two stages in various ways. As shown in section 3.4 the FFT can be implemented more efficiently, in a radix-4 structure, if the number of channels is a power of 4. For the first channelizer this depends also on the time multiplexing factor, as the FFT is performed serially in each of the time multiplexed samples, that are then combined in a small parallel FFT and the power of 4 requirement applies to each parallel FFT. A mixed radix (radix-4 plus one stage of radix-2) FFT does not, however, cause a large penalty and can be considered if there are other reasons to favor a particular factorization.

For demonstration and clarity, in this report we will assume N=512 and 1024 channels for the first and the second stage channelizer respectively. The general considerations on oversampling, filter shape, resolution and total resources usage do not depend on this choice.

3.1.1 Channelization requirements for the LFAA beamformer

For the LFAA, and assuming that the beamforming is performed in the frequency domain, the number of frequency channels in the first stage channelizer determines the phase accuracy of the input signals of the beamformer, and therefore the accuracy in the synthesized station beam. For an antenna at the maximum distance from the station phase center (18 m), at a zenith distance of 45°, the geometric delay is $\tau_{max} = 41$ ns. The maximum phase error at the edge of each frequency channel is $\Delta\phi = \pi\Delta f\tau_{max} = \pi B\tau_{max}/N$, where B is the digitized band and N the number of channels in the first channelizer. For 512 channels over 400 MHz $\Delta f = 0.78$ MHz and $\Delta\phi$ corresponds to up to 0.1 radians, producing a decorrelation of 1.3%. The decorrelation is averaged over the station dishes. Assuming uniform spacing, the average decorrelation is $\approx (\Delta\phi)^2/8$ or 0.13% for the example case. This decorrelation is a function of frequency, and is zero at each channel center, increasing to the above value at each channel edge.

More important is the decorrelation due to the FFT cyclicity: a phase slope across the bandwidth does not produce a delay, but a rotation of the samples in each FFT frame. A number of samples are then placed at the wrong side of the FFT frame, and do not correlate. The fraction of *lost* samples is $\tau_{max}B/N$, i.e. 3% in the previous example. Again this must be averaged over the station, reducing it to about 1.3%, independently from the frequency. Increasing N reduces the decorrelation for both effects.

3.2 Overlapping in the first stage channelizer

In a dual stage filter-bank the input band is first divided into *coarse channels*, and then each *coarse channel* is divided into *fine channels*. Both coarse and fine channels are re-sampled at a reduced sample rate, to keep the amount of information almost constant. The edge of each coarse channel is then affected by aliasing from adjacent channels, and if the input of the channelizer is a real quantity the aliased signal has also conjugate phase, producing a large phase error in the measured cross spectrum. It is thus necessary to provide some guard

space at each coarse channel edge, that depends on the steepness of the channel filter response. Fine channels that fall in the guard space must be discarded and not correlated.

A simple way to overcome this problem is to re-sample the coarse channelized signal at a slightly higher sample rate, thus extending the alias free region. If the alias free regions of two adjacent channels are contiguous, with an overlap between the Nyquist bands of the coarse channels, no fine channel needs to be discarded. The higher sample rate requires more computations per sample and reduces the final spectral resolution, so it must be kept to a minimum. On the other hand, to produce a small oversampling, a steeper channel shaping filter is required, so a tradeoff must be found.

The channel shaping polyphase filter must be flat over a passband with a width equal to the channel separation, and must reject by at least 60 dB any signal from adjacent channels beyond the guard space, i.e those that would be aliased back in the passband.

Oversampling is done in the polyphase filter that precedes the FFT block. The filter convolves a block of sampled data and folds it to the FFT block length (WOLA: Weigth, Overlap and Add operation). The oversampling is done by moving the convolution mask by a number of samples smaller than the FFT block length, and by performing the convolution/FFT processing at a sufficient high clock rate, not to loose input samples.

The relevant filter quantities are shown in fig. 3. The input signal is assumed real, with sample rate f_s and total bandwidth $f_s/2$. The bandwidth is divided into N channels of useful bandwidth, and channel separation, equal to $f_s/2N$. The channelized samples are computed at a higher rate, given by an oversample factor o_f, and the output Nyquist band is thus given by $f_n = o_f f_s/2N$. The filter cutoff frequency is fixed, at half the channel separation, $f_1 = f_s/4N$, while the stop band begins at the frequency $f_2 = f_n - f_1 = (2o_f - 1)f_s/4N$

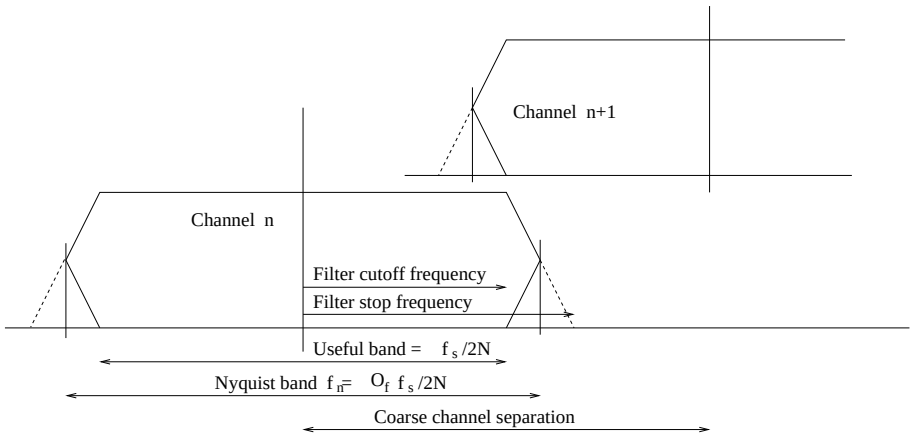


Figure 3: Oversampled, overlapping coarse channels

Each channel will be further divided into fine frequency channels in the correlator, using a FFT with a number of channels that is a power of two. To allow to re-grid these fine channels on a regular grid, the oversampling factor o_f must be of the form $2^n/k$, with n and k integers. Practical values are $32/26 = 16/15$, $32/27$ and $32/28 = 8/7$. In this way only the central $1/o_f$ fine channels are processed, and fine channels from adjacent coarse channels produce a contiguous, regular coverage of the input band.

The resulting low-pass cutoff frequency and stop frequency for the above oversampling factors is reported in table 2. We computed equiripple filters with in-band ripple of ± 0.15 dB, and out-of-band rejection of 60 dB, to evaluate the number of required taps, also reported in the table. The number of required real multiplications per sample (with the filter design described in section 3.2.1) and the data rate increment are also reported. The number of channels N, refer to the positive frequency channels only, as the input signal is assumed real.

Each FFT is performed every $2N$ samples, and therefore the number of multipliers in the polyphase block, equal to the number of required taps divided by $2N$, is independent on the FFT length.

3.2.1 Filter design

Several possible algorithms can be used to compute these filters.

o_f	Cutoff freq.	Stop freq.	N. of taps	N. of mult.	Data rate incr. (%)
32/26	$0.25/N$	$0.3654/N$	$24N$	12	23.1
32/27	$0.25/N$	$0.3426/N$	$30N$	15	18.5
32/28	$0.25/N$	$0.3214/N$	$36N$	18	14.3

Table 2: Oversampling factor and related polyphase filter characteristics

For the second stage filter (see section 5.1 we used a boxcar $(\sin(x)/x)$ filter windowed using a Dolph-Chebyshew window. This produces a very flat passband response and is computationally effective also for large filter sizes. This method is not appropriate for the first stage filter, as it produces very long filters, with an over-specified passband flatness.

We then used the Remez algorithm aiming at an equiripple response, as this allows to independently select the passband and stop-band ripple. The filter has a ± 0.15 dB ripple, and about 62 dB stop-band attenuation. However the algorithm does not converge for filter lengths of more than about 2000 taps. To overcome this problem, the filter has been initially computed for a channelizer with 64 channels, and then filter taps were interpolated, using a spline interpolator, to the final number of channels[5]. The procedure produces a non equiripple response, with slightly lower stop-band attenuations for the first adjacent channels, and much higher (up to 80 dB) attenuation in the interpolated region. The filter impulse response is symmetric, with an even total number of taps. This procedure is described in more detail in a separate memo (in preparation), and in [5].

The normalized filter responses for the filters corresponding to the three oversampling factors listed in table 2 are shown in fig. 4.

Matlab functions to compute the filter taps, and format them either as plain ASCII files or VHDL packages, have been developed and extensively used.

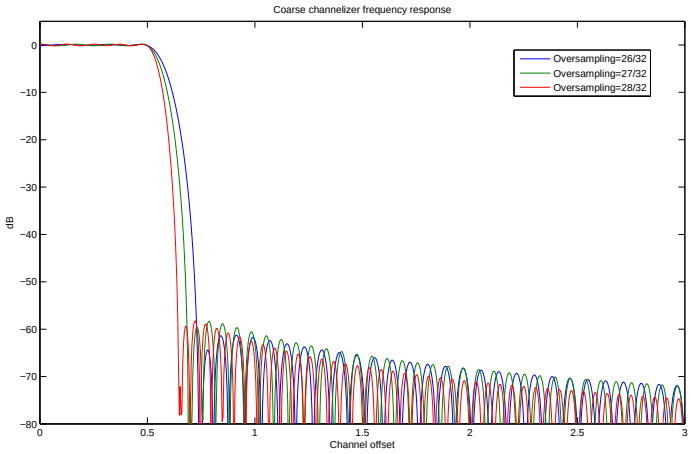


Figure 4: Normalized response for filters with oversample factor of 32/26 (blue), 32/27 (green) and 32/28 (red)

3.3 Polyphase filter structure

The polyphase filter performs the sum $P(b, kM) = \sum_a S(kM + aN + b)F(-aN - b)$, with $b = 0 \ldots (N-1)$. It accepts a continuous set of input samples, for convenience framed in frames of length M, and produces frames of N samples. It is composed of a set of identical blocks, each of which implements a delay of N samples, a memory containing N filter tap coefficients and a multiplier (fig. 5). Each delay is implemented as a dual port memory block. The first block (*delay1*) uses two different clocks for reading and writing, the others use the

same clock, but "rewind" the delay by $N - M$ samples at each frame. Frames are delimited by a *start-of-frame* (SOF) signal that marks the first sample in each frame. The output of all blocks are multiplied by the tap coefficients and summed together.

During a frame, M samples are written into the *delay1* block, and N samples are read from it. Each frame overlaps with the previous one by $N - M$ samples. Thus the output clock rate is faster than the sample rate by the factor $o_f = N/M$. The SOF pulse is transferred to the oversampled clock domain using a safe dual-clock circuit.

The oversampled part of the filter, i.e. everything past the first rate change FIFO, can be run either synchronously, with a clock derived from the sampler clock using a PLL, or asynchronously, with a clock frequency slightly higher than the actual oversampled clock. In the latter case the processing is done in units of an integer frame of N samples, and each sample is validated by a *data valid* signal. Idle clock cycles are inserted between output frames.

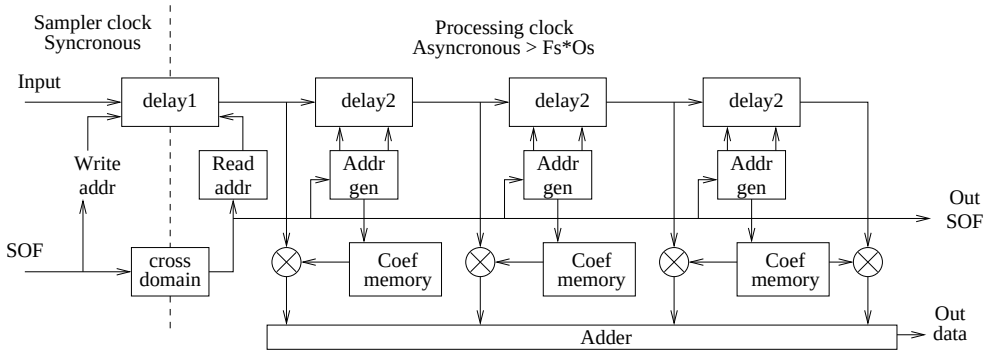


Figure 5: Polyphase filter. The filter is composed of an input block, crossing frequency domains, and successive blocks that use the output clock rate. At each frame period, M samples are written in the first block and N samples are processed from each block. $N - M$ samples are reprocessed in successive frames

3.3.1 Time multiplexing

For all SKA channelizers, the input sample rate is higher than the clock frequency attainable in currently available FPGAs. Sample are thus presented in time multiplexed format, i.e. N_m consecutive sample are processed in parallel at a correspondingly reduced clock rate. Usually N_m is a power of 2. The structure of the polyphase filter does not change significantly. If N_m divides both M and N, as is usually the case for the examples shown above, each parallel sample is processed by a small polyphase filter, with the subset of filter tap coefficients corresponding to these samples. The number of required multipliers is increased by a factor N_m but the memory requirement is the same. The output of the filter is again time multiplexed, and must be processed by a serial-parallel FFT.

An example for $N_m = 4$ is shown in fig. 6. The input signal, at a $4\times$ clock rate, is time multiplexed into 4 parallel data streams, with consecutive samples in each stream. The SOP pulse is transferred to the oversampled cross domain to ensure that all parallel time multiplexed filters operate synchronously among them.

Several optimizations can be used in the FPGA implementation. As all address generators are identical, they can be shared among memory blocks. The memory size, especially for high time multiplexing, is typically too short for efficient memory usage. Some synthesizers can merge together similar memory blocks, but this can more easily be accomplished by using the same block for parallel filters. If more than one signal is processed in parallel, the coefficient memories can also be shared among the filters for the independent signals.

3.4 FFT block

The FFT block converts the (real) polyphase output to a sequential stream of (complex) channelized samples. The block computes the transform of a block of $2N$ samples at a time, but, due to symmetry, only N FFT frequencies need to be computed. This can be performed in several ways. One possibility is to up-convert

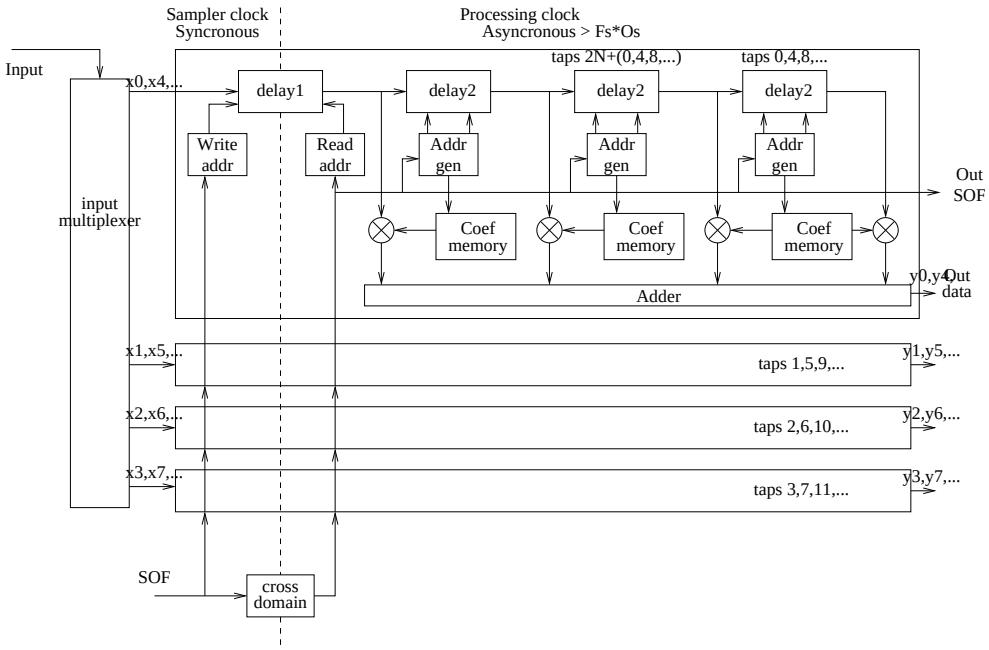


Figure 6: Polyphase filter for 4× multiplexed samples. Each filter processes one parallel sample, implementing 1/4 of the total filter taps

the input signal by 1/4 the sampling rate, filter the result with a low-pass filter and decimate it by a factor of 2. The processing is then done on the complex signal with halved sample rate. The up-conversion is quite straightforward, as the exponential at $F_s/2$ has only the values $\pm 1, \pm i$, but the filter requires a significant number of multiplier resources.

The algorithm adopted here is to compute the transform of a sequence of N complex values $S_{2j} + iS_{2j+1}$, and then combine channel j and $N - j$ to form the actual real FFT outputs at these two frequencies, as described in section 3.4.1.

The first operation is easily done in a time multiplexed sample stream, by just combining successive time multiplexed samples.

The most multiplier efficient FFT algorithm uses radix-4 decomposition, requiring three complex multipliers per stage. The serial/parallel FFT is thus based on a radix-4 serial FFT block designed for the Uniboard library. The FFT basic block accepts 4 complex inputs, and perform 4 FFT in N clock cycles. Each FFT input represents 2 consecutive samples, so the FFT engine can be used to process up to 8 independent real time multiplexed phases and/or independent signals, using a total of 3 integer multipliers every 2 input signals/phases.

For the LFAA correlator $N_m = 4$, so a FFT block can process two independent signals in parallel. For the MID correlator time multiplexing factor can range from 4 to 16, requiring up to 2 parallel FFT units for each signal.

An alternative is to use a commercial IP for the FFT, as the FPGA vendors provide highly optimized blocks for their particular architecture.

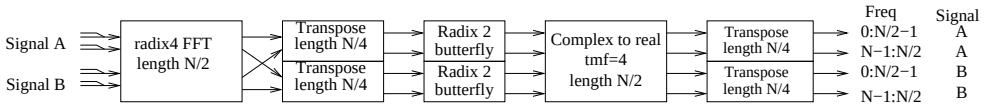


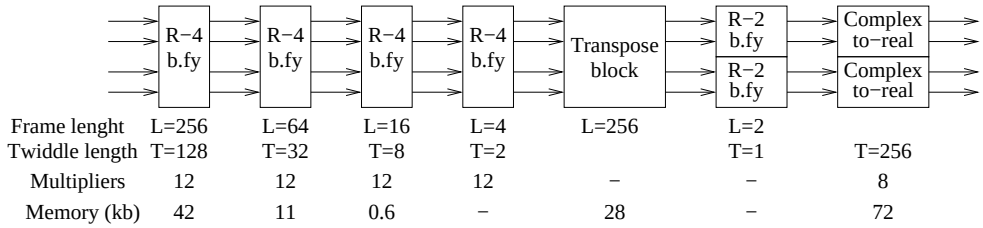
Figure 7: Real input radix 4 FFT, for a frame length of N samples ($N/2$ spectral channels)

The structure of a N point FFT block for a 4-times multiplexed real signal is shown in figure 7. The 4 parallel samples from the polyphase filter are combined into 2 complex signals, for a total frame length of $N/2$

complex samples. The FFT engine is based on a serial radix-4 FFT block, that operates simultaneously on 4 complex data streams. One such block is then used to analyze 2 real time multiplexed signals. The first section computes the first 4 stages of a DIF FFT, with appropriately different twiddle factors for the odd and even streams. The outputs from this first section are presented sequentially for the 4 inputs, and must be reordered in a transpose block in order to have two consecutive samples available simultaneously

These samples are combined in a final radix-2 block (a simple butterfly stage), obtaining two times-2 demultiplexed FFTs for 512 complex inputs. These outputs correspond to four frequency channels spaced by $N/4$ frequency steps. The two time multiplexed inputs (e.g. two polarizations from one station) are still presented sequentially.

The FFT outputs are placed in a final reordering memory, that extracts at each instant the FFT outputs for frequencies f, $f + N/4$, $3/4N - f$ and $N - f$. Combining together these values produce the corresponding FFT outputs for the corresponding real valued input samples. A final reordering block produce two streams, relative to the first and second half of the frequency channels, for each of the two input signals.



Frame lenght	L=256	L=64	L=16	L=4	L=256	L=2	
Twiddle length	T=128	T=32	T=8	T=2		T=1	T=256
Multipliers	12	12	12	12	–	–	8
Memory (kb)	42	11	0.6	–	28	–	72

Figure 8: Resource usage for a 4x time multiplexed FFT, processing 2 real data streams with 512 spectral channels

In figure 8 the resources (memory and multipliers) used in each block are also listed for the case $N = 1024$ (512 frequency channels).

The structure of a single radix-4 stage is shown in fig. 9.

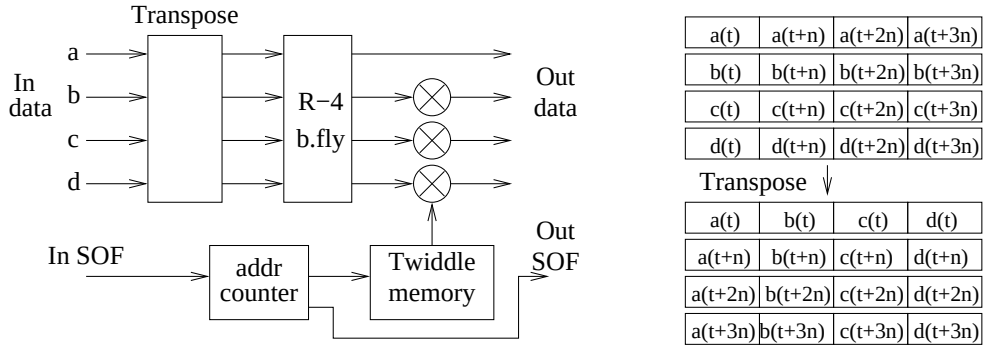


Figure 9: Radix 4 block. Is composed of a transpose block, that aligns samples separated by n positions, a 4-point FFT and a set of complex multipliers

A similar architecture can be used also for $N_m = 8$ and $N_m = 16$ (figure 10 and 11).

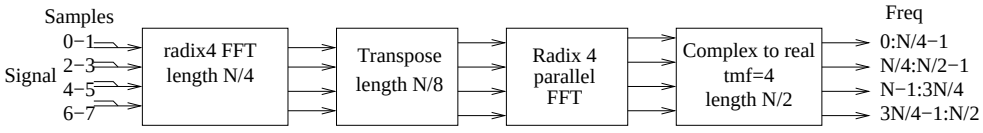


Figure 10: Real input FFT for 8x and 16x time multiplexed signals

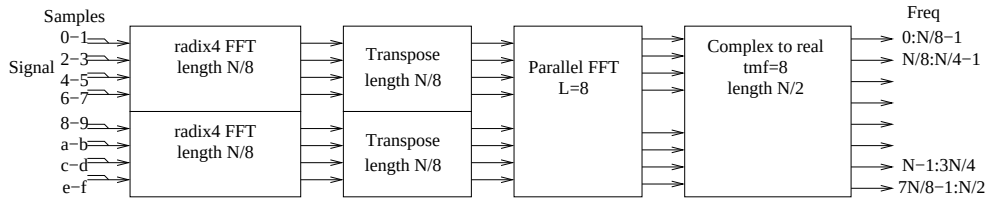


Figure 11: Real input FFT for 8x and 16x time multiplexed signals

3.4.1 FFT of a real sequence

Several algorithms exist for producing a FFT of real valued data, that reduce the number of required operations by about a factor of 2. One can see for example the `rfft()` routine in Numerical Recipes[1].

The basic idea is to compute the FFT of a sequence $y_j, j = 0 \ldots N/2-1$ derived from the original time series $x_j, j = 0 \ldots N-1$, with $y_j = x_{2j} + ix_{2j+1}$. If Y_k is the transform of y_j, with

$$Y_k = \sum_{j=0}^{N/2-1} y_j \exp\left(2\pi i \frac{2jk}{N}\right) = \sum_{j=0}^{N/2-1} (x_{2j} + ix_{2j+1}) \exp\left(2\pi i \frac{2jk}{N}\right) \tag{1}$$

Considering that $Y_{-k} = Y_{N/2-k}$, and denoting with a^* the complex conjugate of a, one can easily show[1] that

$$X_k = \frac{1}{2}\left(\left(Y_k + Y^*_{N/2-k}\right) - iexp\left(2\pi i \frac{k}{N}\right)\left(Y_k - Y^*_{N/2-k}\right)\right) \tag{2}$$

$$X_{N/2-k} = \frac{1}{2}\left(\left(Y_k + Y^*_{N/2-k}\right) + iexp\left(2\pi i \frac{k}{N}\right)\left(Y_k - Y^*_{N/2-k}\right)\right)^* \tag{3}$$

$$\tag{4}$$

The complex to real structure is shown in figure 12. A dual port memory performs the bit reversal operation and, at the same time, reverses the order of the second half of the frequency channels. Channels i and $N/2-i$ are combined together in a first butterfly, with the second input conjugated. The difference is then multiplied by the twiddle factor and the result is combined in a final butterfly to get the two frequency channels i and $N/2-i$, relative to the input real signal. Note that the factor of $1/2$ present in equation 2 is not applied, as it would introduce rounding errors.

The particular case with $i = 0$ is dealt in a special circuitry. In this case frequency 0 and frequency $N/4$ channels are processed in a simpler, and straightforward way.

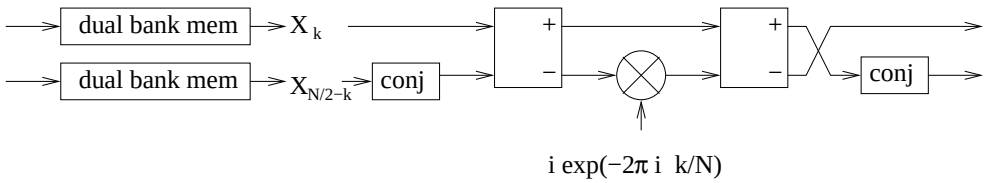


Figure 12: Complex to real conversion stage. The dual port memory performs bit reverse reordering, with ascending and descending order for the upper and lower sequences, respectively. The butterfly-twiddle-butterfly block separates the FFTs for the two real valued inputs

A similar structure has been used for different time multiplexing factors, from 2 to 16. The output of the FFT block is always organized in blocks of 4 (for TMF ≤ 4) or multiple of 4 frequency sections, with each section holding $1/4$ (or $1/4n$) contiguous frequency channels. Each pair of sequences is processed by a processing chain of fig. 12, according to formula 2. The output channels are presented in direct order for the first half of the channels, and in reversed order for the second half.

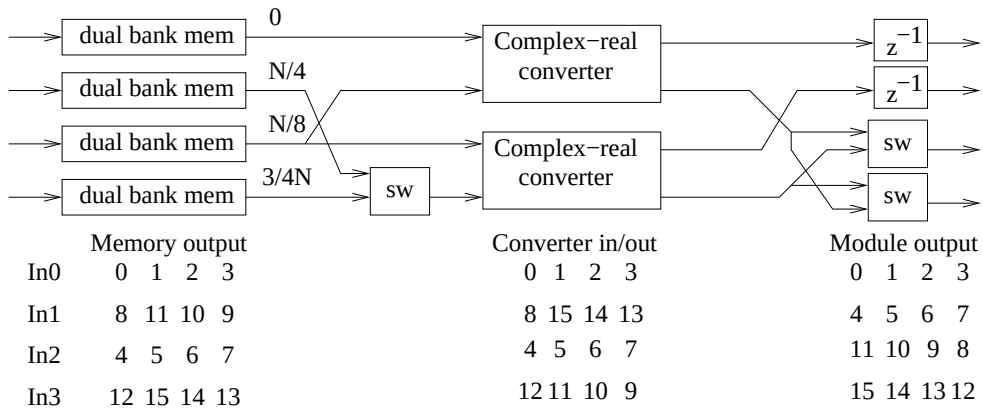


Figure 13: Complex to real conversion stage for time multiplexed FFT. Two blocks are used for pairs of frequency channels. Frequency channel order for the four data streams are show at each processing stage

The different order for the first frequency channel in each time multiplexed stream means that this channel must be processed slightly differently from the others.

An example is shown, for $N = 32$ (16 channel channelizer), in fig. 13. The first group processes channels 0 and $N/4$ separately. The second group processes inputs 2 and 3 in the first cycle, and inputs 2+1 in the remaining part of the sequence. Outputs from the complex-to-real units are in a slightly inconvenient ordering. "Low" channels are ordered correctly, but "high" channels have the first element in each sequence misplaced. The "low" sequences are then delayed by one channel, and the first channel in each 'high" sequence is stored and positioned in its correct place at the end of the appropriate sequence.

3.5 Bit growth and rounding

The channelizer must be sufficiently robust to avoid a signal overflow for any possible input signal. The worst case is a pure tone, that will end up in a single FFT channel, with an amplitude growth of a factor of $N\sqrt{2}$. Gaussian noise increases by a factor of $\sqrt{N}$, but for non white noise the frequency dependent spectral variation must also be considered. For reasonable frequency dependence of the spectrum, noise amplitude in individual channels may vary by up to ± 5 dB with respect to the level corresponding to white noise, so about one extra bit of resolution must be provided to account for these variations.

Input samples are 4-10 bit wide. The polyphase filtering don't significantly change the signal statistics, so the filter output can be re-quantized to 10 bits without losing significance. The FFT growth adds 11 bits, that exceeds the available 18 or 19 bit multiplication size, so an intermediate rescaling is required along the FFT chain. A possibility is to have a fixed rescaling of the signal after 3 radix-4 stages, dividing the intermediate product by a factor of 8.

3.6 RFI detection, total power, re-quantization

The coarse channelizer output is composed of complex samples, represented by two integers with 18-25 bit accuracy (tbd). Sample accuracy must be reduced to fit in the limited physical channel bandwidth interconnecting the two stages. Bandwidth is also limited for the memory required for corner turning.

A total power meter is thus present at the output of the channelizer, producing a coarse power spectrum of the input signal. Two spectra are computed, one with a time resolution of ≈ 1 ms, that is used for automatic transient RFI excision, and one with a much longer (0.1 to 1 second) integration time, used for gain adjustment and for static RFI flagging. Both integration times are user selectable, and their exact values must be determined considering likely RFI statistics.

Before further processing, samples are:

- re-quantized using variable thresholds, automatically set using the (slow) total power measurements

- flagged as RFI if the total power exceeds some predefined value. The whole coarse channel is semi-permanently flagged as invalid on this condition
- temporarily flagged as RFI if the fast total power measurement exceeds the slow measurement by a predefined factor

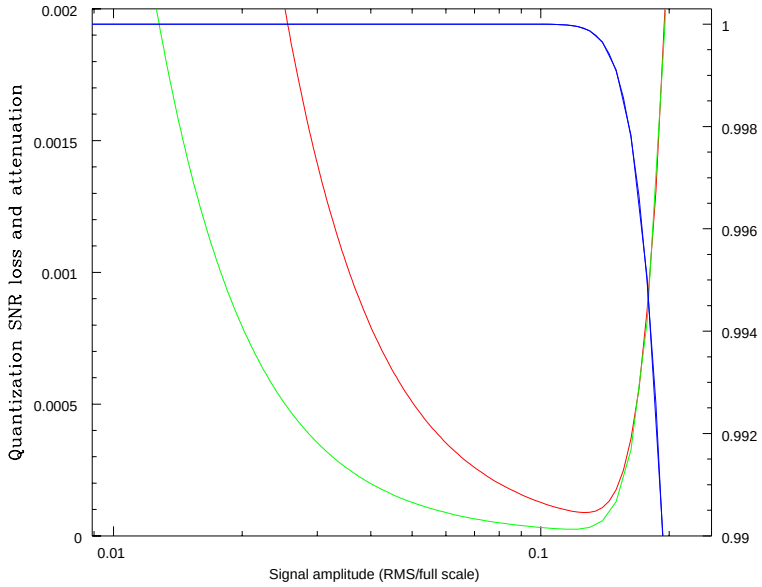


Figure 14: Gain (blue) and quantization loss for a 8 bit (red) and 9 bit (green) quantization, as a function of the signal RMS level σ divided by the quantizer total range $\alpha\sigma_e$

Some coarse channels at the band edge are not processed at all, because they fall in the analog filter aliased zone, or are outside the receiver band. The processed bandwidth is then smaller than the ADC sampled band, and this reflects in the total number of effective correlated channels and final channel resolution.

Re-quantization introduce excess noise, roughly Gaussian with RMS amplitude of $1/\sqrt{(12)}$ the sample step, and nonlinearities, due to truncation. Both these effects are shown in fig. 14, for 8 and 9 bit signal representation, as a function of the signal RMS amplitude (in units of the total representation range). Added noise is within 0.2% even with a RMS signal amplitude of 6-7 units. Nonlinearities became important when the signal amplitude exceeds 1/10 of the total range, i.e. 25 units for 8 bit quantization, but are very small up to 1/8 of the total range (32 units for 8 bit quantization). This limits to 12–14 dB the dynamic range of the coarse channels, i.e. the power in a narrow line (totally contained in the channel) cannot be more than 15-25 times the average spectral density in the channel. To achieve the required 20 dB dynamic range specification at least 10 bit re-quantization is required. In this case we can have a wideband signal of 8 units RMS against a narrow band signal of 100 units RMS, with 99% of the power in the latter.

3.7 Coarse channel corner turning

A polyphase filter-bank produces frames of samples relative to a single sample time, and all frequency channels. Subsequent frames refer to subsequent sample times, with a frame rate f_c equal to the original, un-channelized time resolution f_s divided by $2M = 2N/O_f$: $f_c = f_s/2M = f_s O_f/2N$. The number of coarse frequency channels that is kept for processing is $N_c = B/f_c$, where B is the actual processed bandwidth (e.g. 300 MHz for the LFAA).

Usually a number n_s up to several signals are processed in parallel in the coarse channelizer, producing a set of bidimensional arrays (frames) of samples, of dimension $n_s \times N_c$, at a frame rate $f_s/2M$.

In the case of the LFAA, the coarse channelizer processes all antennas in a station (or in a station tile), and then combines all antennas in a station in a single signal. In this case the final $n_s = 2$, for the two polarizations in a station. This situation is shown in figure 15: the channelizer/beamformer output consists in a total of 4 streams, for 2 frequency channels (in the order described in chapter 3.4.1) and 2 polarization. Each stream is composed of 192 coarse channels (150 MHz of input bandwidth, 64 channels are discarded at each end of the digitized band), and all samples for one sample time are output together in one block. A total of 600000 samples per coarse channel are produced in one integration time. The example can be easily adapted to other channelizer architectures, with different time multiplexing and/or number of signals processed in parallel.

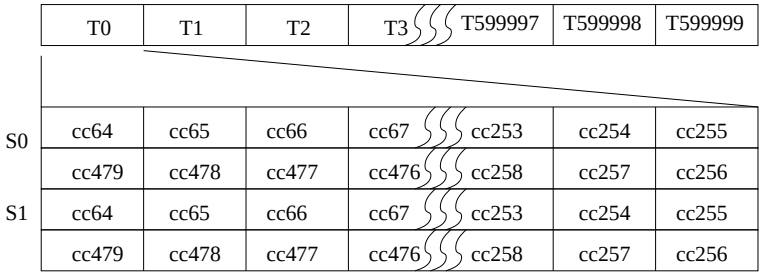


Figure 15: Output ordering of the samples from the coarse channelizer. The example refers to the LFAA channelizer/beamformer but applies, with different numbers, to other architectures. cc(n) refers to coarse channels, T(n) to sample times, S(n) to signals

The second stage channelizer must process a single continuum sample stream for each coarse channel, with an initial dead time of several thousand samples required to fill the polyphase filter. For efficiency is then necessary to process very long sequences, and a good choice could be to process a whole elementary (minimum) correlation time at a time. If multiple second stage units are instantiated, the corresponding number of coarse channels and/or independent signals can be processed in parallel. Samples must then be rearranged to provide a number of parallel streams, with all samples in one integration time arranged consecutively in each stream. This *corner turning* operation is performed in an external memory, by writing the samples form the coarse channelizer in (almost) their natural order, and reading back them in the order required in the second stage channelizer.

Most large memories (DDR3, DDR4, HMC) work most efficiently with a record length of the order of 64-128 bytes. For example DDR4 records can be written/read without dead times, synchronously changing the record address every 64 bytes and thus allowing a random access of memory records of this size. Data must then be packed in records of this length, using a format that allows to reconstruct the output data streams in the required format. An example of such a format is shown in fig. 16, again for a possible implementation of the LFAA. Each record stores samples for 8 coarse channels, 2 stations and 2 consecutive sample times. A total of 48 such records are required to store two coarse channelizer output frames (sample times). Records are stored consecutively into memory in a circular buffer. If the buffer is long enough, it implements also a transient event capture functionality.

Records are read back in consecutive sample order, i.e. one record every 48. The samples are reassembled into 8 separate streams, that are routed to 8 independent fine channelizers.

The resulting overall conceptual structure of the corner turner is shown in fig. 17.

4 Delay compensation

The signal at each antenna arrives with a time varying geometric delay. The delay includes both the difference in arrival time at the antenna with respect to the array phase center, and the propagation delay through the interconnection between the antenna and the CSP. For an antenna maximum distance of D, and assuming the worst case of a source at the horizon, the maximum delay is $\tau = D/c + D/c_f$, with c_f the signal speed along the interconnecting fiber, $\approx c/1.6 = 0.63c$. Then $\tau \approx 2.6D/c$. The delay τ must be multiplied by the sample rate to translate it in a number of samples. The three arrays have a similar maximum delay, of about 500 k-samples for the low and survey arrays and of about 2 M-samples for the Mid array.

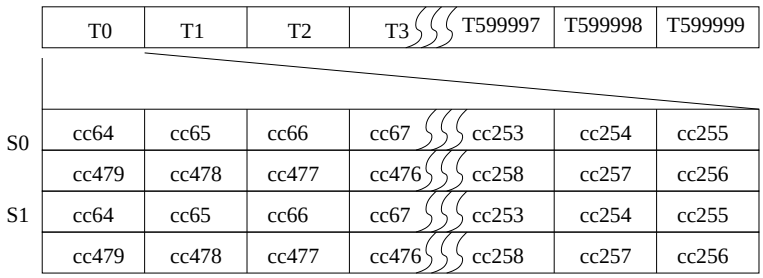


Figure 16: Format of a corner turner memory block. The example refers to the LFAA channelizer/beamformer. cc(n) refers to coarse channels, T(n) to sample times, S(n) to signals

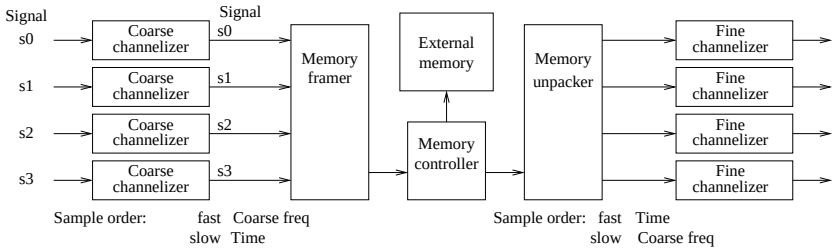


Figure 17: Conceptual architecture of the corner turner memory

Delay is compensated in two steps: the larger part of the delay is compensated using a delay FIFO, that is either set for the approximate delay at the center of the elementary integration period, and then left fixed, or dynamically adjusted as the delay changes. The delay variations across the integration period are compensated using a phase slope in the frequency domain. Positive and negative delays must be accommodated, thus requiring twice this number of samples in the delay buffer.

Maximum delay rate is $\dot{\tau} = \tau\omega_e$, were $\omega_e = 7.27(-5)\text{s}^{-1}$ is the Earth diurnal angular velocity, resulting in a delay rate of a few samples during an integration period. For the LFAA the maximum delay residual on half integration time is 5.8 ADC samples, while for the MID correlator at the maximum sampling rate it is 18 ADC samples.

Residual delay is corrected using a phase rotator in the frequency domain, adding a time varying phase to each coarse and/or fine frequency channel. The phase is equal to the instantaneous delay times the nominal center frequency of the channel $\phi(\nu, t) = \tau(t)\nu$. Delay correction in the frequency domain produce a decorrelation equal to the number of samples in the delay divided by the FFT length, as a phase slope in the frequency domain correspond to a rotation, not a shift, of the FFT input frame. If the coarse delay is corrected during the integration, the maximum residual delay is limited to $\pm 1/2$ sample, and the corresponding decorrelation is very small. This requires careful propagation of the time discontinuity in the polyphase delay blocks, and add some complexity to the filter design.

If the delay correction is performed after the first channelization, the delay variation during an elementary integration is $\ll 1$ sample, and the coarse delay can be adjusted only between integrations.

If local oscillators are present in the signal chain (as is the case for the MID and survey correlator), the frequency must include the local oscillator frequency ν_{lo}. Contribution to the phase change due to the term $\dot{\tau}\nu_{lo}$ can vary quite fast, up to several turns across the integration period. If the corresponding phase variation across a single fine channelizer frame introduces a non-negligible decorrelation, this requires a phase correction both before and after the fine channelizer. This is discussed in more detail in section 4.3.

4.1 Delay compensation in coarse filter-bank

Delay correction can be performed either in the coarse and in the fine channelizers. Performing it in the coarse channelizer has the advantage that the data stream at the channelizer input is already corrected for at least the

gross delay. This allows an easy derivation of a separate signal stream for the PSS/PST subsystems, or for a VLBI beamformer. The amount of buffer memory is however quite large, up to several M-samples.

Delay variation during a single elementary integration period can be quite large, ±5.8 samples for the LFAA, and up to ±18 samples for the MID array. The survey array, being relatively compact, has a maximum delay variation of ±2 samples. This produces a potentially large decorrelation, in the LFAA and MID correlators, up to several %, if the gross delay is not corrected dynamically. This, in turn, require that the coarse filterbank be aware when a delay change has occurred, to properly perform the WOLA operation.

Fine delay correction can be performed after the first channelizer, or after the second channelizer. Correcting the phase after the first channelizer is slightly simpler, as it can be done in the coarse channelizer, but leaves a slight phase slope across each coarse channel, up to a fraction of a degree. Although small, this has to be corrected to prevent systematic effects in the visibility phases.

4.2 Delay compensation in fine filter-bank

For the LFAA and Survey correlators, data arrive at the CSP already coarse channelized. Delay correction must therefore be performed in the fine polyphase filter-bank.

Delay range in samples is ≈ 1000 times smaller than for the raw samples, as the coarse channelized data rate is ≈ 1000 times slower. A FPGA internal memory block can thus be used to implement the delay buffer. The residual delay to be corrected is correspondingly larger, but considering the high number of channels (256-1024) in the fine channelization, the residual phase error within each fine channel is less than ±0.35 degrees, with a negligible decorrelation. Due to the much larger time step, the delay variation across a single integration is negligible and the coarse delay can be adjusted only between integrations.

If the coarse channelized data are used to synthesize a VLBI beam at the correlator delay center, with delays corrected only to ±1/2 the coarse channelized data rate, the residual delay errors produce a significant decorrelation and beam distortion. This can be mitigated using a separate fine delay correction scheme, or by re-synthesizing a coarse channel from the fine channelized data after fine delay correction. This is possible in theory if the polyphase filter response satisfies the the Princen-Bradley criterion for invertibility, but almost doubles the complexity of the fine channelizer.

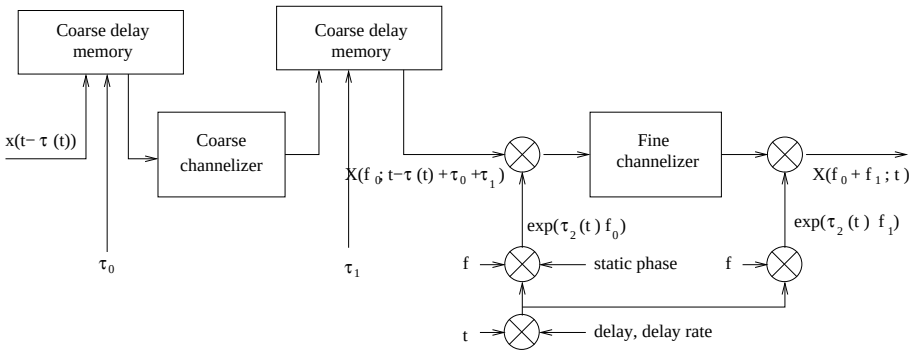


Figure 18: Delay correction for delay performed after coarse channelization. Integer sample delay is corrected in a FIFO memory. Fine (subsample) delay and fringe rotation is corrected after

4.3 Fringe rotation and fine delay corrections

The delay rate produces a phase variation with time that is proportional to the signal sky frequency, i.e. it includes any local oscillator present in the signal processing scheme. This phase variation must not change significantly during the period in which the phase correction must be kept constant. This period is basically the sampling rate of the signal to which the correction is applied, i.e. either the sampling rate at the output of the coarse or the fine channelizer. The coarse channelized data rate is fast enough not to produce any decorrelation, so the problem is whether the fringe rate can be safely applied on the fine channelized data. This would have

the advantage of applying a single phase correction both for the time dependent and frequency dependent components of the delay.

The fringe rotation can be relatively fast, and corresponds to a small but significant fraction of a turn in a fine channelized sample. If the fringe correction is performed after fine channelization, the varying phase across the FFT frame causes a decorrelation. For a standard (non polyphase) FFT, the amount of decorrelation is $D = 1 - sin(\Delta\phi)/\Delta\phi$, where $\Delta\phi = \pi\nu_f/\Delta f_f$ is the phase variation in one fine sample, ν_f is the fringe frequency and Δf_f the fine sample rate (or sample width).

As the polyphase channelizer actually process samples convolved with a filter, the whole response of the filter to a phase-varying signal must be considered. With good approximation this corresponds to a fixed increase in the phase variation, $\Delta\phi = \alpha\pi\nu_f/f_f$, with α slightly greater than 1. A good approximation for the decorrelation is thus

$$D = \frac{1}{6}\left(\frac{\alpha\pi\nu_f}{\Delta f_f}\right)^2 \tag{5}$$

α has been evaluated both analytically and by numeric simulation, and the relation 5 proved correct up to $\Delta\phi = 0.2$ for a wide variety of filter shapes. For the filters described in chapter 5.1 α ranges from 1.08 to 1.26 and 1.43, for a polyphase filter of 5, 7 and 9 taps respectively. The corresponding decorrelation increases from 16% to about a factor of 2.

The relevant parameters, for a power-of-2 channelization with reasonable assumptions on the oversampling factor and on the bandwidth of the digitized signal. are reported in table 3. The worst case (9 taps) decorrelation is reported in the last column of the table.

	Max distance (km)	Delay rate (ns/s)	Max freq. (GHz)	Fringe freq. (Hz)	Channel width (Hz)	Decorr. 9 taps (%)
LFAA	100	24.3	0.35	8.49	904	0.03
MID band 1	150	36.4	1.05	38.2	2720	0.07
MID band 2	150	36.4	1.76	64.0	4360	0.07
MID band 3	150	36.4	3.05	111.	7630	0.07
MID band 4	150	36.4	5.18	188.	10900	0.10
MID band 5	150	36.4	13.8	502.	21800	0.18
Survey	50	12.1	1.50	18.2	1130	0.09

Table 3: Fringe frequency and associate decorrelation if fringe correction is performed after fine channelization

For the low correlator, phase decorrelation would be about 0.03%, and therefore no separate correction for the fringe rrotation is required before the fine channelizer. For most of the bands of the Mid, and for the Survey correlators the decorrelation is still less than 0.1%, that is likely acceptable in a reasonable repartition of the correlation processing losses. The only situation where the decorrelation begins to be significant is band 5 of the Mid correlator, for which it approaches 0.2%.

5 Second stage polyphase filter-bank

The second stage channelizer must produce N channels, non overlapping, with good channel-to-channel insulation.

5.1 Filter design

Each channel is sufficiently narrow to have negligible phase variation due to uncompensated delay, and to resolve most of astronomic spectral lines. It is likely, however, to have significant variations in the signal level among adjacent channels. In this case the filter shape should preserve the integrated flux density over a set of channels, i.e. the response to a monochromatic signal, summed among adjacent channels, should be independent form the signal frequency, within specifications. For example, if the signal frequency is exactly at a channel boundary,

the two adjacent channels should measure half the power. i.e. the 3 dB point of the filter should occur close to the channel nominal boundary. With this prescription, the filter satisfies with very good approximation, even if not exactly, the Princen-Bradley criterion for invertibility: it is *constant overlap-add* (COLA), i.e. the impulse response of the filter is constant independently of the impulse position in the input frame.

Channel insulation is required not to contaminated unrelated channels when a strong signal (astronomic or RFI) is present in the received data. At least 60 dB of insulation is specified in level 1 specifications [2]. For immediately adjacent channels the integrated rejection over the channel is specified as better than 30 dB. The specification is however in conflict with the previous one (uniformity) unless the filter transition region is $\approx 1/100$ of the channel width. A series of filters, requiring k (complex) multiplications per sample have been computed, and the relevant parameters listed in table 4. All frequencies are expressed as a fraction of the channel width, referred to the nominal channel boundary. For example the first filter ($4N$ taps, 4 complex multipliers) has a -6 dB and the -60 dB (start of the nominal stop-band) points at about 21% and 59% respectively of the adjacent channel, and an ideal signal, with a rectangular spectrum matching the channel width, will produce a $-11.6dB$ response in the adjacent channels. Broadening is expressed as the second moment of the filter power response, normalized with that of an ideal rectangular filter.

Filters have been computed using a $\sin(x)/x$ filter multiplied by a Dolph-Chebyshew window, a much faster computational method with respect to the Remez algorithm. The Remez algorithm produces an improvement in the spillover of about 17%, or 0.7 dB, that is marginal for the present discussion.

N. of multipliers	-6dB freq.	Stop freq.	Stop-band att. (dB)	Spillover (dB)	Broadening (%)
4	0.1037	0.590	-11.6	-61.0	26.8
5	0.0830	0.444	-12.7	-72.3	17.2
6	0.0690	0.394	-13.6	-60.7	11.9
7	0.0593	0.327	-14.3	-72.0	8.7
8	0.0517	0.296	-14.9	-61.0	6.7
16	0.0258	0.147	-18.0	-61.0	1.6
32	0.0129	0.574	-21.0	-60.0	0.3

Table 4: Second stage polyphase filter characteristics

The filter response for the filters in table 4 with 5 to 32 multipliers, and and the total response (sum over adjacent channels) for a monochromatic line, for the filters with 5 to 8 taps are shown in figure 19.

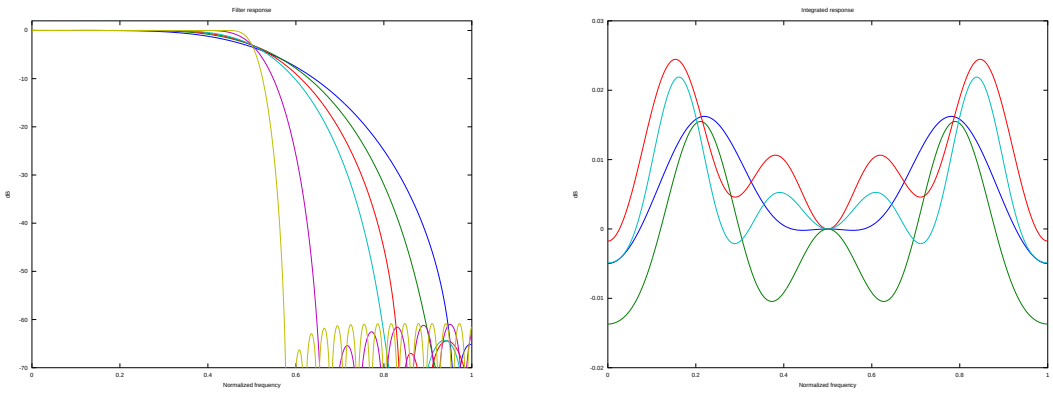


Figure 19: Response in dB of the 2^{nd} stage channel filter, for 5 (blue), 6 (green), 7 (red), 8 (cyan), 16 (purple) and 32 (yellow) multipliers per sample. Left: single channel response. Right: response integrated over adjacent channels (5 to 8 multipliers)

The integrated response uniformity is comparable for the filters, and always better than 0.025 dB (0.6%). To

achieve a spillover compatible with the -30 dB specifications, about 500 (complex) multiplications per sample are required in the filter. Considering that a radix-4 FFT with 512 spectral points require ≈ 20 (real) multiplications per sample, this requirement adds a significant amount of complexity to the channelizer. The less demanding requirements listed in section 1.1 can be satisfied even with just $5N$ taps. The stop-band attenuation for the same parameters is somewhat better for an odd number of multipliers.

The 5 multiplicators filter design produces an increase in effective channel width of 17%, and requires about the same number of multipliers in the filter and FFT sections. The response for all channels in the filter-bank, referred to a single channel, is plotted in figure 20. The filter nominal band is blue, adjacent filters are black and green. Typical rejection is around -70 dB, and is always below -60 dB except for small portions of the adjacent channels and for the spillover. Any monochromatic signal cannot contaminate more than 2 adjacent channels, as per the specifications. If multiplication resources are not constrained, a 7 taps filter would produce a slight increase (8%) in spectral resolution. More complex filters do not appear to be justified.

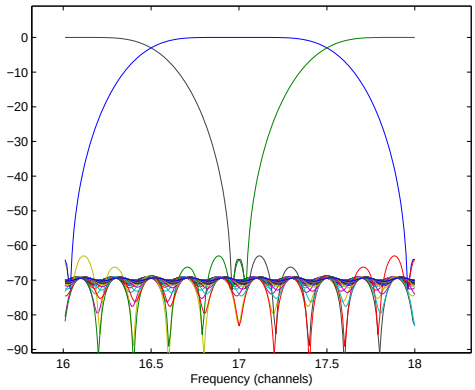


Figure 20: Response in dB of all the filters in the filter-bank, as seen from a single filter (#17)

.

5.2 Physical implementation

The second stage channelizer uses a straightforward polyphase structure with complex inputs. The FFT uses a radix-4 architecture, analogue to that shown in fig. 8, without the final complex-to-real section. The FFT block can perform 4 FFT in N clock cycles, so it can analyze simultaneously 4 coarse frequency channels. The output fine frequency samples are masked as invalid if they fall outside the coarse channel boundaries, and not sent to the X processor.

Input samples are complex, using an integer representation with 8-12 bit accuracy (to be determined).

6 Data representation

Signal is represented by a real or complex time series, with a number of bits ranging from 6 to (tbd, likely ≈ 20) across the computation chain.

Less than 6 bits representation is discouraged, both because of intrinsic high quantization noise and non-linearity. A 5 bit representation has either a quantization noise of ≈ 0.35%, but high nonlinearity, or good nonlinearity over a very limited range, but a quantization noise exceeding 1% [6, 7]

Invalid samples can be present anywhere in the sample stream, e.g. due to RFI. It is therefore useful to have a dedicated sample code for invalid samples. The maximum negative number (sign 1, mantissa 0) is particularly suited to this purpose, as it only slightly reduce the allowable data range, and moreover it preserves symmetry between positive and negative values, removing a potential source for DC bias. Therefore it has been adopted as a standard across the SKA CSP.

For complex quantities, it is sufficient to mark as invalid the real part of a sample. The imaginary part can have any value, e.g. an error code marking the particular reason of invalidity. The code can then be used to gather statistic informations. This however adds extra complexity, as the code must be copied during processing.

Data is processed in frames, that correspond to one processing unit. Input frames for the polyphase filter bank have a length of one FFT operation, with one output sample per frequency channel for each input frame. Frames for the correlator represent a whole elementary integration time. Usually framed data have some extra service lines, at least a *start-of-packet* and a *data valid* line. This latter has a different meaning form the *invalid* code, as it marks clock cycles without any data, that do not correspond to any actual physical time, while the *invalid* code marks an actual sample, with an associated time value, that happens to contain corrupted data.

Different policies can be used in a processing block to deal with invalid samples in the input frame.

- The simplest and more general policy is to substitute invalid samples with zeros, and count them. The output frame is invalidated (all output samples substituted with the *invalid* code) if the count exceeds a given threshold. A threshold of 1 invalidates the output frame if *anyone* of the input samples is invalid. This processing is required in the channelizer, where the output depends on all the input samples. Different thresholds can be used for the central, more important, portion of the polyphase filter and for the frames affecting the filter wings, to prevent a single bad sample to affect a very long portion of the data.
- Processing is performed only on valid samples. A counter counts the number of processed samples, in order to normalize and weight the result. This is appropriate for correlation. This has the advantage of allowing a high percentage of the input samples to be invalid, at the cost of increasing the output size, as the counter must also be transmitted with each output value.

Framed data contain also some ancillary informations, in a frame header. This header is transmitted together with the data between processing units, using a self describing format like VDIF or SPEAD. A minimum set of informations specified in the header includes the following:

- Station. Station code includes polarization, as separate polarizations are treated like independent signals. For example, the station code could be an even number, with the LSB of the code reserved for polarization.
- Beam number. Multiple beams must be supported in all telescopes. Beam number affects, among others, the delay processing.
- Time. Time can be expressed in sample unit from the beginning of the integration period. Delay correction is not included in time count, i.e. a particular sample is assigned a different time before and after gross delay compensation. Time step is different after each channelization stage, so the time step is also included in the header information.
- Frequency and frequency step.

Each quantity must be specified with an initial value, a dimension and a size, as usually a mix of samples with different attributes (e.g. multiple stations, polarizations and beams, plus multiple time values) are present in a single frame.

Informations from the header can be extracted at a processing module input, implicitly used inside the module, and reformatted at the output, without being physically carried along. For example data from different stations can be stored in different addresses in a buffer memory.

References

[1] "Numerical Recipes, the art of scientific computing", chapter 12, "FFT of a single real function"

[2] B. Carlson: "SKA CSP internal requirements definition for chiplevel and chip memory level design and development start", SKA CSP Memo 001

[3] "SKA CSP Level 2 Requirements", document SE-2_SKA-TEL_CSP_SE_SSA_SRS_001

[4] G. Comoretto: "Data processing in the ALMA Tunable Filterbank", ALMA document CORL-60.01.07.00-013-B-MAN

[5] G. Comoretto: "A design method for very large FIR filters", Arcetri memo series 3/2012

[6] G. Comoretto: "Quantization of Gaussian noise and RFI immunity", SKA-CSP memo n. 8 (2014)

[7] J. Bunton: "Correlator bit allocation", SKA-CSP memo n. 9 (2014)