Mem. S.A.It. Vol. 88, 154

Memorie della

Radioastronomic signal processing cores for the SKA radio telescope

G. Comoretto, S. Chiarucci, and C. Belli

Istituto Nazionale di Astrofisica – Osservatorio Astrofisico di Arcetri, Largo E. Fermi 5, I-50125 Firenze, Italy, e-mail: `comore@arcetri.astro.it`

Abstract. Modern radio telescopes require the processing of wideband signals, with sample rates from tens of MHz to tens of GHz, and are composed from hundreds up to a million of individual antennas. Digital signal processing of these signals include digital receivers (the digital equivalent of the heterodyne receiver), beamformers, channelizers, spectrometers. FPGAs present the advantage of providing a relatively low power consumption, relative to GPUs or dedicated computers, a wide signal data path, and high interconnectivity.
Efficient algorithms have been developed for these applications. Here we will review some of the signal processing cores developed for the SKA telescope.
The LFAA beamformer/channelizer architecture is based on an oversampling channelizer, where the channelizer output sampling rate and channel spacing can be set independently. This is useful where an overlap between adjacent channels is required to provide an uniform spectral coverage. The architecture allows for an efficient and distributed channelization scheme, with a final resolution corresponding to a million of spectral channels, minimum leakage and high out-of-band rejection. An optimized filter design procedure is used to provide an equiripple response with a very large number of spectral channels.
A wideband digital receiver has been designed in order to select the processed bandwidth of the SKA Mid receiver. The receiver extracts a 2.5 MHz bandwidth form a 14 GHz input bandwidth. The design allows for non-integer ratios between the input and output sampling rates, with a resource usage comparable to that of a conventional decimating digital receiver.
Finally, some considerations on quantization of radioastronomic signals are presented. Due to the stochastic nature of the signal, quantization using few data bits is possible. Good accuracies and dynamic range are possible even with 2-3 bits, but the nonlinearity in the correlation process must be corrected in post-processing. With at least 6 bits it is possible to have a very linear response of the instrument, with nonlinear terms below 80 dB, providing the signal amplitude is kept within bounds.

Key words. Signal processing – Radioastronomy – Square Kilometre Array

1. Introduction

Data processing in modern radio telescopes is increasingly performed using digital techniques. The astronomic radio signal is converted to a digital form often just after the amplification, without a frequency conversion that is performed digitally. Whenever this is not possible, the analogic signal processing chain includes only a single frequency conversion, to adapt the signal bandwidth to that of available high speed ADCs.

The processing system must therefore be able to deal with very large input sampling rates, up to several tens of GHz, and with a large number of input signals. As large telescopes are typically located in remote sites, power generation is also a concern, and power saving architectures must necessarily be used.

Data processing is composed of a set of relatively standard operations, like frequency translation, filtering, frequency channelization, and beamforming, usually in a fixed configuration. The computation chain is thus relatively simple, but requires a large number of operations. For example the down-conversion of a signal requires about 20 operations per sample, or 200 Gop/s for a signal sampled at 10 gigasample per second. Interferometric instruments are particularly demanding, as the signal from each antenna must be correlated with all the other ones. An interferometer with hundreds of antennas may require computing capabilities of the order of several hundreds of peta-op/s.

For all these considerations systems based on custom or programmable logic are favored. Systems based on commercial computers, although very flexible, present excessive power consumption, and GPU based solutions are advantageous only when complex algorithms are required, like for example pulsar search surveys. Field programmable logic arrays (FPGAs) are currently the preferred processing engines, as they use relatively low power, have a high granularity well suited for massive parallel processing, and have dedicated circuitry for a large number of high speed interconnection links. FPGAs, unlike dedicated logic (ASIC), can be used in general purpose processing boards not linked to a specific application, and can be reprogrammed to correct for errors or to add new functionalities to an existing instrument.

The Square Kilometre Array is a large interferometer, with a final collecting area of the order of one million square meters, spanning the frequency range of 50 MHz to 14 GHz. In the initial phase it will be composed of two instruments, for about 1/20 of the final collecting area. SKA Low is composed of 131,072 log periodic antennas in the frequency range from 50 to 350 MHz, and will be built in Australia, while SKA Mid, composed of 192 dishes in the frequency range between 0,7 and 14 GHz, will be built in South Africa. A third instrument, already built in the same Australian site, is composed of 16 antennas, each equipped with a phased array feed, i.e. a plane of about 200 individual antennas that are digitally combined in order to produce a focal plane radio imaging detector. In all these cases the digital processing system, due to the high number of input signals, their large bandwidth or both, must be able to analyze tens to hundreds of terasamples per second.

For this telescope we developed a set of functional blocks organized as a general purpose library. The library has been developed trying to maximize the generality and reusability of the functional blocks, as described in section 2. In part this library reuses general purpose blocks developed in the Radionet Uniboard FP7 program.

Some of these modules are described in detail: a polyphase oversampling channelizer, in section 3.1 and a beamformer, in section 3.2, for the Low telescope, and a down-conversion module, in section 4, for the Mid telescope. A general analysis about the quantization effects on the linearity of the data processing is briefly outlined in section 5.

2. FPGA firmware development strategy

As these signal processing operations are relatively common, several libraries, both commercial and developed by the astronomic community, currently exist. In particular we examined:

– Proprietary graphic generation tools. These are provided by the FPGA vendors, and include a rich set of basic functional blocks that can be used to generate a high level processing unit as a Simulink model. The tool then automatically translates this model to the code used to actually program the FPGA. These tools are however not very efficient, using typically 20–50% more FPGA resources with respect to direct coding. They are also bound to a specific FPGA vendor, and it is not possible to migrate the design to a different vendor.

– The Casper project. This is a wide effort, carried on by a consortium of several radioastronomic institutes, to develop both a set of hardware platforms and a library of high level building blocks, with the typical functionalities required in a radioastronomic instrument. It is built above the proprietary Xilinx graphic development tool, and is thus bound to FPGAs from this vendor. Board development is a slow process, and boards are typically 2-4 years behind the current state of the art. Casper boards have been used in the Italian radio telescopes (Bartolini 2016).
– OpenCL, is a C-like language, originally developed for programming GPU boards. Altera has provided a version of the language for their FPGAs. It however requires a specific board support package for each board, and the FPGA must have some form of local intelligence, and/or a direct link to a host computer. It is also very specific to the Altera vendor.

Considering the limitations of these tools, in particular the difficulties in transferring a design to a different vendor, we chose to describe the modules in the VHDL hardware description language. This general purpose language allows to describe the processing in a way that is relatively similar to a standard programming language. Aspects like signal mapping, timing, etc. must be explicitly coded, and this makes the language difficult to use. It is in general not possible to code without at least some knowledge of the underlying physical structure, but there are consolidated programming techniques that allow at least some level of abstraction.

To facilitate high level design we adopted a general coding template for each module with a standard external interface. Modules are linked together in a very simple, and standardized structure, to build the data processing application. We used the AXI4 standard, developed by the ARM consortium, both for signals, represented as AXI4 streaming interfaces, and for the control system, with each module implementing an AXI4-lite slave.

The FPGA contains only minimal local intelligence, or nothing at all. The modules composing the processing chain are remotely programmed using a AXI4-over-Ethernet interface, implemented either as a state machine or as a simple (and fixed) program in a synthesized microcontroller. The remote master sends commands, setting and reading registers inside the FPGA, as UDP packets. The protocol guarantees that each packet has been correctly received, both for the commands to the FPGA and for the responses to the master. The AXI4 slave interface is automatically generated from a register description in an XML file.

The top level code for each module contains this interface and the custom functional core, that has a more specific, but interface independent, port structure. Adapting a module to a different interface is thus relatively easy. In fact several modules were imported from a similar library, developed as part of the Uniboard Joint Research Activity of the FP7 Radionet project, by just changing the interface layer that, in the Uniboard, is based on the Avalon bus.

Modules do not assume a particular FPGA family. Any vendor dependence is dealt by encapsulating the relevant low level function in a vendor agnostic module. By linking the library containing the correct low level files it is possible to port the module to a different vendor/family.

Modules are heavily parametrized, in order to be reused in different situations. For example parameters for the channelizer module include the input and output data width, the number of channels and the time multiplexing of the input signal.

As an example, we were able to port a spectrometer design (Comoretto et al. 2006), initially developed for one specific Altera board, to a Xilinx based system (Comoretto, Melis 2010), without changing the relevant VHDL code.

To aid the module design we used extensively the SysML system description language (Belli 2016).

3. The SKA Low station processing system

The SKA Low telescope is composed of 131,072 individual cross dipole, log periodic antennas, sensitive to radio waves from most of the sky. Antennas are grouped into *stations* of 256 elements each. Each station is equivalent to a dish with a physical diameter of 35 m, with a receiver bandwidth spanning from 50 to 350 MHz.

The conceptual data processing chain is shown in figure 1. Signal from each antenna is channelized into channels about 0.8 MHz wide. Then 256 antennas are aligned together by applying a time varying phase to each channel and summed together to form a *station signal*.

The resulting signals are further channelized to a final frequency resolution of up to 400 Hz, and corresponding channels from each possible pair of the 512 stations are correlated to produce $\approx$ 128 K (frequency dependent) visibilities. These in turn are integrated, Fourier transformed and deconvolved to produce a sky image.

Processing up to the formation of station signals is performed in a distributed processor composed of 8192 board with two Kintex Ultrascale FPGAs each. This tile processing board (TPM) has been developed by Sanitas srl, in collaboration with the Italian SKA group, and is described elsewhere in this workshop.

Signals from the stations are combined into visibilities in a central signal processor (CSP), also based on Xilinx FPGAs. Visibilities are converted into images in a cluster of conventional processors, the Science Data Processor (SDP). All the elements in the system are interconnected by a dedicated high speed Ethernet network (currently 40 Gbps, 100 Gbps in the final version), using SPEAD, a standard radioastronomic UDP-based data protocol (Manley et al. 2010).

The Arcetri group has been involved in the design of the whole station data processing chain. The block structure for this firmware is described in figure 2. The blocks for the 40 Gb Ethernet, the ADC JESD204 and the external DDR memory interfaces are standard commercial modules. The UDP formatter and the monitor and control subsystems have been designed by the UK SKA group. The remaining modules have been developed by the Arcetri group. These modules are quite general, and can be adapted for other instruments.

The firmware is implemented into two Kintex Ultrascale FPGAs. Each FPGA processes the signals from 8 antennas, 2 polarizations, up to the channelization and beamforming. Then the signals from all 16 antennas is combined together, with each FPGA processing half the total bandwidth. 16 TPM boards are daisy chained using a standard network switch to produce a single signal from 256 antennas, that is then sent to the tile processing module.

The main blocks, described in more detail below, are the channelizer and the beamformer. Other blocks are used for calibration and diagnostic functions:

- A test signal generator, that allows to test the signal processing chain using a pseudorandom Gaussian noise generator and up to two sinusoidal tones.
- A integrating cross spectrometer, that can produce the power spectrum of all signals, and the cross-correlation of two signals, with the frequency resolution given by the channelizer
- A data capture module, used to send to the SKA calibration subsystem samples from all antennas, one spectral channel in a predefined time interval.

3.1. The polyphase filterbank

The frequency resolution required for SKA is very high, of the order of 10^6 channels in the *zoom mode*. Frequency channelization is performed using FFT based algorithms, that are limited to a few thousand points in FPGAs without resorting to large external memory. The channelization is then performed into 2 steps, of $\approx 10^3$ points each.

To control the channel shape, the Fourier transform is preceded by a particular form of a finite impulse response filter, in the so called

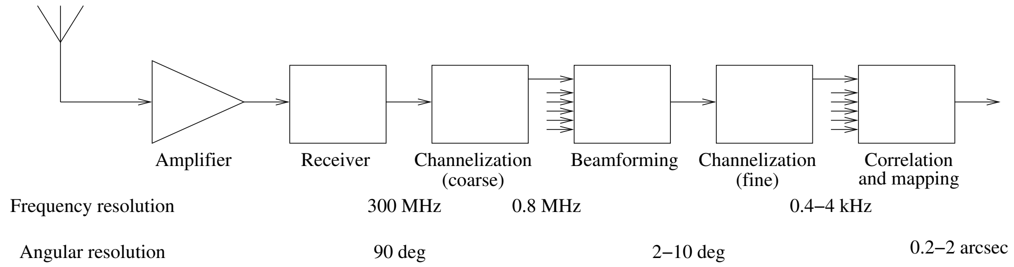

Fig. 1. Processing in the SKA Low telescope. At each stage either the spectral or the spatial resolution is increased. The final product is a three dimensional map, with the frequency on the third axis. Four maps are actually produced, containing also the full polarization information

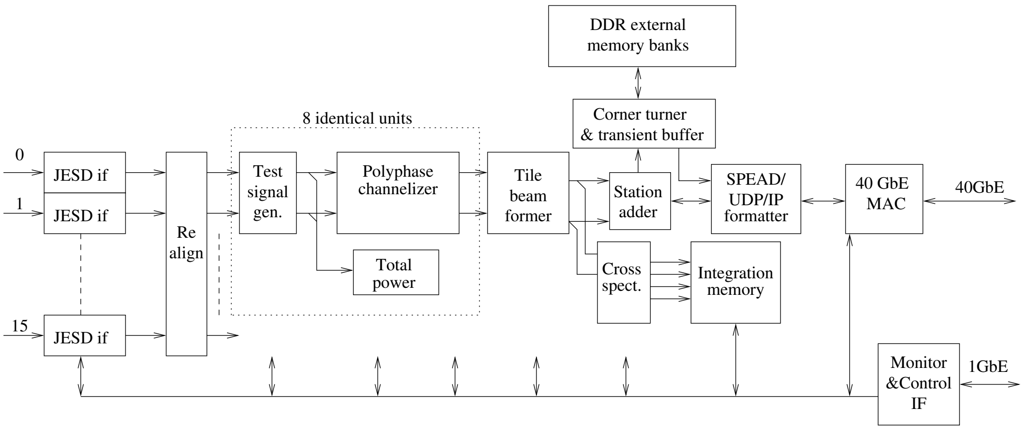

Fig. 2. Firmware for the Tile Processing Module

polyphase filterbank (PFB) architecture described e.g. in Harris et al. (2003). This allows to obtain a very flat passband, and a high rejection of unwanted signals (Radio frequency interferences, RFI). This is particularly important, as most of the input power seen by each antenna is represented by anthropogenic radio emissions that must be effectively excised.

At the edges of each band the filter must necessarily have a finite transition region, that cannot be processed, due to aliasing, if the channel separation is equal to the channelizer output sampling rate. The currently available PFB implementations, however, have this limitation, resulting in *holes* in the final frequency coverage.

For SKA a complete frequency coverage is essential for wideband spectral surveys. A modified PFB version has then be developed, in which the output sampling frequency and the channel separation can be set independently. This is done by rewinding by a small factor the input data stream between Fourier transforms, producing an oversample of the output channels.

The module has also been optimized for real valued, time multiplexed data. Due to limitations of the FPGA speed, the input signal, sampled at 800 MSps, is multiplexed into 4 parallel streams at 200 MSps. The PFB core then operates on this representation. Increasing the time multiplexing, up to a factor of 32, it is possible to process signals with a bandwidth of up to 4 GHz. The block diagram of the time multiplexed, oversampling filter is shown in figure 3.

The main FFT block uses a proprietary radix-4 FFT core, that processes in parallel 4 complex, or 8 real signals. In our case one FFT block processes the two polarizations for one antenna. The design is optimized to minimize memory usage, and the maximum operating frequency exceeds 1 Msample/s for the version adopted in the SKA channelizer.

The filter element of a polyphase filter-bank is the most resource intensive part of the whole system. Minimizing filter length is thus essential. Minimal filter length is usually attained with equiripple design. Equiripple algorithms (e.g. the Remez-McKellan one), however, fail to converge for filters with more than a few hundred elements. In our case the filter length is of the order of 10^4 elements, and the usual approach in these cases is to convolve of a $\sin(x)/x$ function with an appropriate windowing function. This produces filters with very high stop-band, exceeding what is actually needed, at the expense of a increased filter length. We adopted the approach of designing a filter for a smaller channelizer, using the Remez-McKellan algorithm, and interpolating the result to the final required resolution (Comoretto 2012).

In this way we obtained the filter response in fig. 4. The in-band ripple is ±0.2 dB, the stop-band is initially 60 dB and increases to 85 dB after a few spectral channels. The filter order is 14 times the FFT length. For comparison, filters with the same performances designed using traditional algorithms have an order of at least 18 times the FFT length.

3.2. Station beamformer

A beamformer produces an equivalent telescope by aligning the 256 antenna signals in each station for a specific direction in the sky. The alignment is done with respect to a common position for each station, while the alignment between different stations is performed in the correlator.

This alignment can be performed in the frequency domain by applying a phase correction to each frequency channel, $\phi = -f_c\tau$, with f_c the channel center frequency and τ the geometric delay. τ varies with time and, due to the large number of individual antennas, the SKA beamformer must be able to autonomously track it for at least several seconds.

The SKA Low telescope must be able to produce multiple beams, with a constrained total bandwidth, for example to perform accurate pulsar timing by simultaneously observe several pulsars scattered across the sky. Each beam corresponds to an independent frequency region, that can be the same for all beam, partially overlapping or completely different.

Therefore, even if the processing is relatively simple, the SKA beamformer is a relatively complex module. It is described in greater detail in Comoretto (2015).

The module block diagram is shown in figure 5. The channelized frame (i.e. samples for all frequency channels at the same time) is stored in a memory. A structure composed of a set of tables extract from memory the frequency channels corresponding to the individual beams, for up to 8 beams. For each beam and antenna a delay processor computes the geometric delay, using a linear approximation of its time variation. The delay is then multiplied by the channel frequency and the resulting phase converted to a complex phasor using a lookup table. The table is stored only for the first quadrant, to save memory resources.

4. Digital baseband converter

It is often necessary to select a portion of the radio telescope input bandwidth, down-convert it to a band starting near zero frequency (usually called *baseband*), in order to process it with a smaller bandwidth (and sampling rate, for a digital signal). This function can be performed in a digital system by a digital baseband converter (DBBC).

The conceptual schematics of a digital baseband converter is shown in figure 6. The input signal is multiplied by a complex exponential, and low-pass filtered using a complex finite impulse response filter. The result is decimated by an integer ratio, equal to the desired

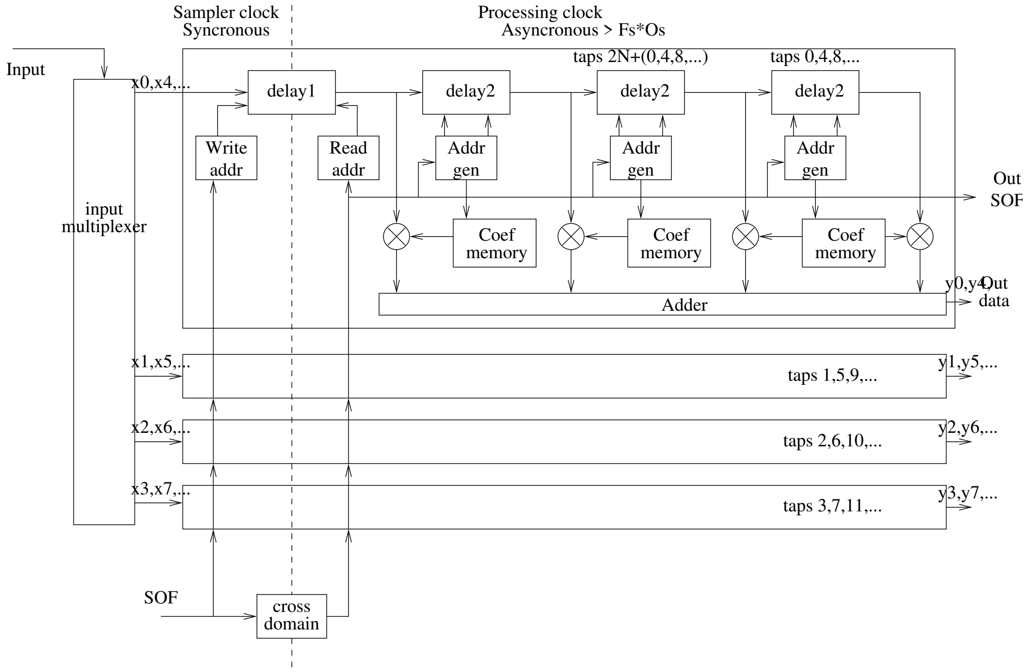

Fig. 3. Polyphase filter for an oversampling, time multiplexed filterbank

final band. If the result must be real valued, it is up-shifted by 1/2 of the band, and the real/imaginary components are interleaved to double the effective sampling rate.

A component that implements this architecture has been developed, and used for example in the spectropolarimeter for the SRT multibeam receiver (Comoretto et al. 2006). The decimating low pass filter can provide a decimation factor from 1/2 to 1/256, in binary steps. The filter uses a fixed number of multipliers, exploiting the longer time between samples to increase the filter order, obtaining the same fractional passband at all decimations.

A more sophisticated filter is required for the SKA band 5 Mid receiver. This receiver is used to select a band of 2.5 GHz, sampled at 5.5 GS/s, from a digitized band of 14 GHz, sampled at 30 GS/s. When the input bandwidth is very high the signal is heavily parallelized, with at least 64 samples for each FPGA clock cycle for this particular signal. In this case a multi-stage architecture allows to reduce the sampling frequency at an early stage, using a very simple filter. A more sophisticate filter operating at the reduced frequency determines the final, high performance, bandwidth. The decimation factor that is required for this filter, with an output sampling frequency of 5.5 GS/s, is 11/60, i.e it is not integer. In this case the low pass filter must also act as an interpolator.

The filter structure is shown in figure 7, and the signal processing is described in figure 8.

The first filter has a complex impulse response, selecting a portion of the input bandwidth that includes the desired region. The filter is very short (40 taps), allowing to save computing resources at this very high sample rate, with a transition region equal to the passband. The decimation process translates the selected band to baseband, but with a frequency rotation, that is corrected by multiplying the signal by a complex local oscillator.

A final filter, with real impulse response, is used for the final band selection and for the interpolation. The interpolation is performed by implementing several filters, with the correct response for each possible interpolating value.

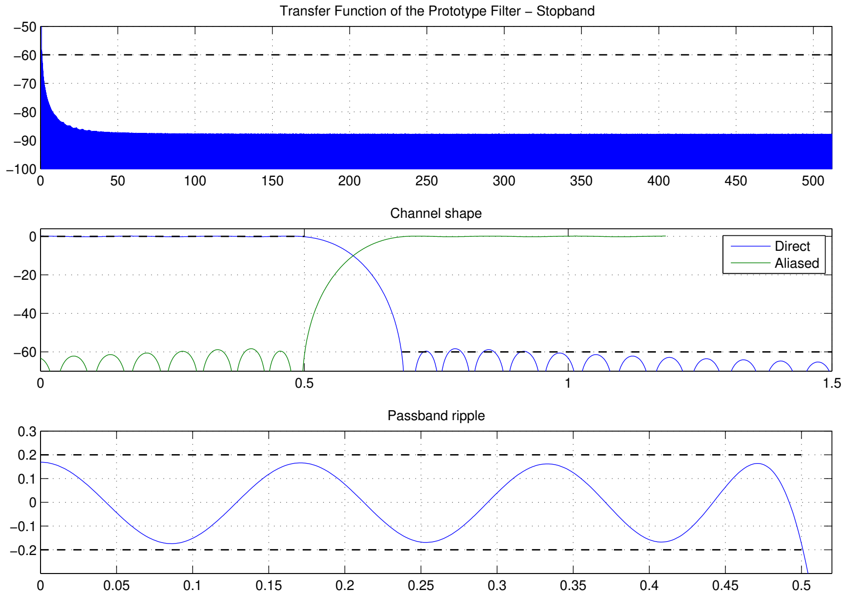

Fig. 4. Filter response for the SKA Low polyphase filterbank. Top: Response over the whole input band. Middle: Zoom over one channel, with direct and aliased response. Low: Passband ripple. Horizontal scale in spectral channels

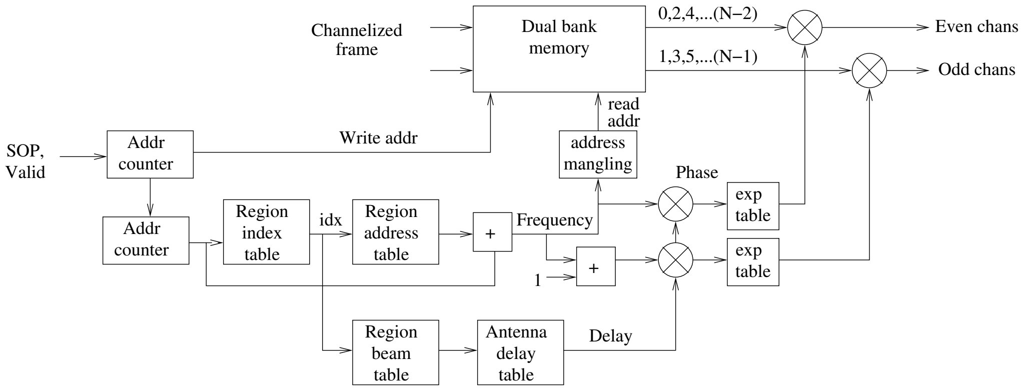

Fig. 5. SKA Low station beamformer

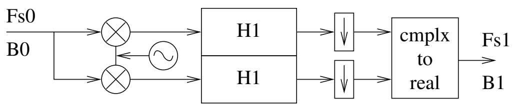

Fig. 6. Standard architecture of a digital baseband converter

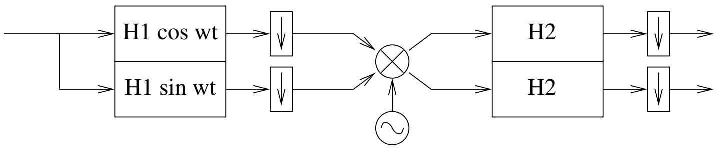

Fig. 7. Structure of a two-stage digital baseband converter. The signal is filtered using a tuned low pass filter H1, decimated, frequency rotated and further filtered by a second low pass filter H2

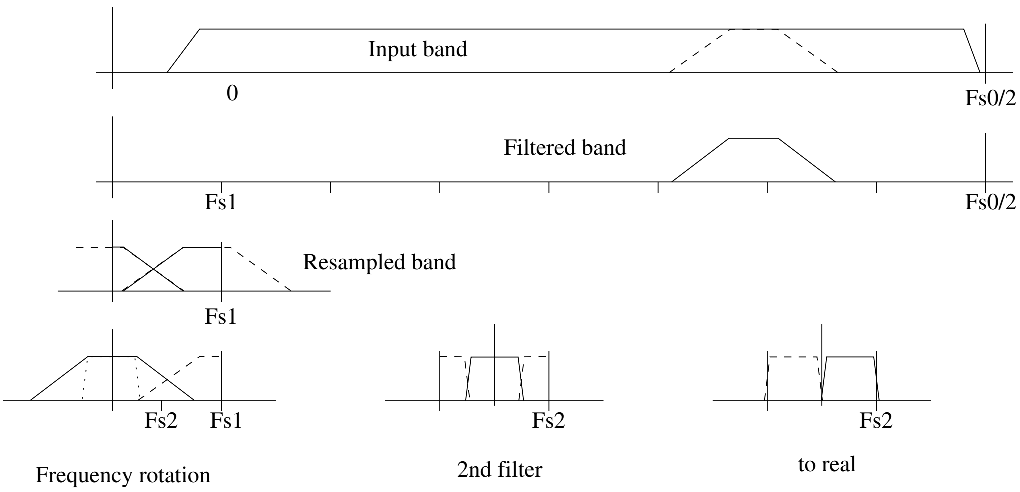

Fig. 8. Signal processing in a two-stage digital baseband converter

Since only one filter is used at each given time, the correct filter is selected simply by choosing the right tap coefficient values, stored in a single memory. To reduce the number of multipliers, the filter uses the clock cycles between two output samples to compute both parts of the current and the next sample. For example in this case only 11 every 15 clock cycles produce an output sample, but the filter uses also the 4 remaining cycles to begin the computation of the next samples.

5. Quantization of radio signals

The radio signal is a stochastic noise, with a Gaussian statistics. The observable is always the product of two signals (or of the signal with itself), i.e. the correlation product of two signals. The two signals can be described using a normal bivariate distribution, characterized only by the covariance matrix, or by the two RMS amplitudes, σ_1, σ_2 and the cross correlation coefficient ρ.

The statistic nature of the signal allows a very drastic quantization, even with just one bit (the signal sign) without a large sensitivity loss. Using a signal representation with 3 or 4 bits, the sensitivity loss is 3.7% and 1.2% respectively.

Data processing is performed on these signals using a larger integer representation, but the result is re-quantized to typically 8-12 bits, due to the limited bandwidth of the interconnection network.

While the sensitivity loss is small, a more serious issue is the linearity of the process. With proper linearization of the correlation product, extremely high dynamic ranges can be achieved, in excess of 100 dB, even using a small number of bits. But in presence of interfering, non stochastic signals, or when the linearization cannot be performed correctly (e.g.

on the intermediate quantizations), quantization linearities may reduce the dynamic range of the final image. The linearization is also very computing intensive, especially if it must be performed on millions of individual visibilities calculated every few seconds. Time variable RFI also poses a problem of dynamic range, especially when the RFI signal can be much stronger than the radioastronomic one. In this case the quantization process must remain linear over a large range of the signal level, i.e. with and without the RFI.

Using a Taylor expansion of the bivariate distribution, and following in detail the quantization and correlation process, we derived an expression for the first nonlinear term in the quantized correlation product for an arbitrary number of quantization bits and input signal level. The detailed analysis is reported in Chiarucci and Comoretto (2015).

The main results are:

- When the two signals being correlated are different, the linearity remains very large (more than 60 dB) if at least 4 bits are used in representing the signal, and if the signal amplitude is comprised between 1 quantization step and 20–25% of the maximum representable integer.
- The added quantization noise is close to the theoretical value of $1/12$, compared to the intrinsic signal noise of σ^2.
- For the product of the signal with itself (autocorrelation), the nonlinearity is significant for any quantization with less than 9 bits. In these cases a linearization is always necessary, but is relatively simple to implement.

The allowed amplitude for the signal to be quantized is thus bounded on the lower side by the allowable added quantization noise, and on the upper side by the allowable nonlinearity. For SKA, the added noise in the intermediate quantizations must be lower than 0.1%, leading to a minimum signal level of 9 quantization steps. For 8 bit representation and at least 60 dB of linearity, the maximum signal level is 33 quantization steps, resulting in a dynamic range of about 11 dB. For the final quantization before the correlation, the allowable noise is 0.5%. Assuming a 6 bit correlator, the quantization correction can be avoided if the signal RMS amplitude is in the range between 4.0 and 8.1 quantization steps.

Acknowledgements. This work is supported by the Italian participation to the SKA project. It is also based on previous work supported by FP7 grant 283393.

References

Bartolini, M., et al. 2017, MmSAI, 88, 172
Belli, C., et al. 2017, MmSAI, 88, 141
Chiarucci, S. and Comoretto, G. 2015, Quantization noise and nonlinearities in the correlation of two Gaussian signals, SKA-CSP memo 0016
Comoretto, G., et al. 2006, A modular multichannel spectrometer – design study, INAF - Osservatorio Astrofisico di Arcetri, Internal Report, 4/2006
Comoretto, G., Melis, A. 2011, Exp. Astron., 31, 59
Comoretto, G. 2012, A design method for very large FIR filters, INAF - Osservatorio Astrofisico di Arcetri, Internal Report, 3/2012
Comoretto, G. 2015, LFAA Tile Beamformer structure INAF - Osservatorio Astrofisico di Arcetri Internal, Report, 2/2015
Harris, F. J., Dick, C., and Rice, M. 2003, IEEE Trans. on Microwave Theory and Techniques, 51, 4
Manely, J., et al. 2010, SPEAD: Streaming Protocol for Exchanging of Astronomical Data, SKA-SA internal memo, 2010/10/07