ASTRON
Netherlands Institute for Radio Astronomy

Impact analysis of change request to use 1 MHz beamlets instead of 0.78125 MHz beamlets in APERTIF and ARTS

	Organisatie / Organization	Datum / Date
Auteur(s) / Author(s): Eric Kooistra	ASTRON	2015
Controle / Checked: Andre Gunst	ASTRON	
Goedkeuring / Approval: Andre Gunst	ASTRON	
Autorisatie / Authorisation: **Handtekening / Signature** Andre Gunst	ASTRON	

Distribution list:

Group:	Others:
Joeri van Leeuwen Wim van Capellen Gert Kruithof Andre Gunst Hajee Pepping	

Document history:

Revision	Date	Author	Modification / Change
0.1	2015-04-15	E. Kooistra	Creation
0.2	2015-04-28	E. Kooistra	Updated planning after review by Hajee, Daniel and Andre.
1.0	2015-04-29	E. Kooistra	Minor update after review by Joeri.

Table of contents:

References:

[1] "Arts Requirements Specifiction", ASTRON-RS-020, J. van Leeuwen
[2] "Apertif System Requirement Specification", ASTRON-RS-006, Rev C, W.A. van Capellen
[3] "Detailed Design of the Digital Beamformer System for Apertif", ASTRON-RP-413, G. Schoonderbeek, A. Gunst, E. Kooistra
[4] "Detailed Design of the Correlator System for Apertif", ASTRON-RP-1400, G. Schoonderbeek, A. Gunst, E. Kooistra, HJ Pepping
[5] "Understanding Digital Signal Processing", 3rd edition, R. Lyons
[6] "Apertif Filterbank Firmware Specification Part 2", ASTRON-SP-054, E. Kooistra
[7] "A serial radix-N FFT VHDL library", nov 2014, G. Comoretto
[8] "On Hardware Implementation of Radix 3 and Radix 5 FFT Kernels for LTE systems", 2011 IEEE, J. Löfgren, P. Nilsson.
[9] "Uthernet interface specification", ASTRON-SP-041, E. Kooistra
[10] "Data path packet interface specification", ASTRON-SP-042, E. Kooistra

Terminology:

Apertif	APERture Tile In Focus
Arts	Apertif Radio Transient System
Beam	Group of beamlets that point in the same direction
Beamlet	Beam formed subband, a small beam spanning one subband
BF	BeamFormer
BN	Back Node
bps	Bits per second
BW	BandWidth
FFT	Fast Fourier Transform
FN	Front Node
FPGA	Field Programmable Gate Array
GbE	Gigabit Ethernet
HDL	Hardware Description Language
Im	Imaginary
IO	Input Output
MAC	Multiply and Accumulate, Medium Access, Monitoring and Control
Nof	Number of
PN	Processing Node (BN or FN)
Re	Real
RF	Radio Frequency
sps	Samples per second
Subband	Frequency band, unit output of the filterbank
X	Correlator

Definitions:

N	1024	FFT size of the FFT in the Apertif BF subband polyphase filterbank
K		Average number of beams per subband
N_{CB}	37	Number of compound beams, N_{CB} = K
P	4	Wideband rate factor of sample clock rate divided by digital processing clock rate
RF_{BW}	400M	Radio frequency total input bandwidth
CB_{BW}	300M	Beam former total output bandwidth per compound beam
B_{sub}		Subband bandwidth = beamlet bandwidth
f_s	800M	Digitizer sample frequency of the ADC at the Apertif BF frontend
f_{clk}	200M	Data processing clock rate in the FPGA
N_{band}	16	Number of bands in the Apertif BF to process the full CB_{BW}
N_{FN}	24	Number of subband per FN in the Apertif BF (= N_{sel}/N_{band})
S	64	Number of ADC signal paths in the frontend of the Apertif BF
S_{BN}	4	Number of ADC signal paths per BN in the frontend of the Apertif BF
nof_uni	4	Number of UniBoards per polarization and dish in the Apertif BF
nof_bn	4	Number of back node FPGAs (BN) per UniBoard
nof_fn	4	Number of front node FPGAs (FN) per UniBoard
$W_{beamlet}$	6	Number of bits per beamlet voltage sample at the output of the Apertif BF
N_{chan}	64	number of channels per beamlet in the Apertif X
nof_adc_bits	8	Number of bits per ADC sample at the input of the Apertif BF
nof_pfb_bits	16	Number of bits per subband sample

1 Introduction

1.1 Purpose

This document provides the information for the Arts change request to Apertif to have 1 MHz beamlets at the output of the Apertif beamformer (BF).

1.2 Scope

This document investigates the impact of changing the current beamlet bandwidth of B_{sub} = 781250 Hz into B_{sub} = 1 MHz regarding both technical and development aspects. The conclusion regarding the technical aspects is that the change to B_{sub} = 1 MHz is quite feasible. Therefore section 2 first describes the impact regarding the development time and planning and then section 3 describes the technical details.

1.3 Background

The Apertif beamformer uses a sample rate of f_s = 800 MHz and the filterbank uses a 1024-point FFT. This results in subbands and beamlets of B_{sub} = 800M/1024 = 781250 Hz. The RF frequency band can be downconverted in steps of 10 MHz relative to 0 Hz. For a standalone radio telescope this 10 MHz frequency grid and 781250 Hz subband resolution are fine, but for a radio telescope that needs to measure together with other radio telescopes it is preferable to have a frequency grid of 1 MHz. Arts science cases SC1 Pulsar timing and SC2 VLBI require combining signals from different radio telescopes [1]. Therefore the Apertif BF beamlet bandwidth needs to be changed to preferrably B_{sub} = 1 MHz. The Apertif BF beamlet output is used by both the Apertif correlator (X) and by the Arts science cases (SC1, 2, 3 and 4).

2 Development impact

2.1 Number of human weeks

Changing to B_{sub} = 1 MHz requires adding the mixed radix feature to the FFT in the filterbank and requires a parameter value change to designs and applications that use the Apertif BF. The parameter value change of B_{sub} and N_{sel} does not require new functionality in the firmware or software, but it does require verification in simulation and validation on hardware. Table 1 lists the development tasks and estimates the number of human weeks that they will need.

Nr.	Component	Task	Sub task	Effort [weeks]
1	N=800-point FFT	• Specification of a mixed radix-2 and 5 FFT (Part 3 of [1]) • Implementation and verification of the mixed radix FFT in the filterbank	DSP/firmware engineer Firmware engineer	8 10
2	Apertif BF	• Implementation and verification of parameter change for B_{sub} = 1MHz and N_{sel}=304 subbands to the VHDL firmware and Python test scripts	Firmware engineer - for BN filterbank - for FN beamformer - for both together	 1 1 2
		• Adapt and verify the existing Apertif BF MAC software (weight calculation, weight control)	Software engineer	2
		• Adapt and verify the existing Apertif BF Python scripts	Hardware engineer Instrument engineer	1 1
3	Apertif X	• Adapt and verify the existing Apertif X VHDL firmware and Python software	Firmware engineer	2
4	Arts	• Adapt and verify the existing Arts VHDL firmware and Python software	Firmware engineer	1
	Total			31

Table 1: Development overview for changing to B_{sub} = 1 MHz

2.2 Planning in time

Assumptions:

- The current development for Apertif BF, X and Arts can still continue with B_{sub} = 781250 Hz and N_{sel} = 384 subbands, but the engineers should even more than before keep in mind that these will change to B_{sub} = 1 MHz and N_{sel} = 304 subband. In this way the parameter change will have nearly no implication on the implementation and only require verification and validation.
- The Apertif BF firmware that runs on the UniBoards can only support either B_{sub} = 781250 Hz or B_{sub} = 1 MHz. It is awkward to maintain two sets of FPGA images because synthesizing these images takes ~3 hours and loading them into the FPGAs also takes ~1 min. Therefore it is best to change to FPGA images for B_{sub} = 1 MHz at some time in the planning and then freeze and no longer support the FPGA images for B_{sub} = 781250 Hz. The best time to do the change to B_{sub} = 1 MHz is when the mixed radix FFT is available and when the Apertif BF, Apertif X and Arts are running stable on B_{sub} = 781250 Hz, because then the change will least disturb the ongoing development.

Task 1 in Table 1 about the development of the mixed radix FFT will take about 4 months. This development needs to be done first but cannot start immediately due to limited resources.

Tasks 2, 3, and 4 in Table 1 concern the transition from B_{sub} = 781250 Hz to B_{sub} = 1 MHz for Aperitif BF, Apertif X and Arts will take about 3 man months to apply and verify.

In total the change takes about 7 man months. Some work can be done in parallel but most tasks need to be done in series.

Remarks:

- Changing B_{sub} takes development time but it will also have some beneficial side effects. First we will then have a mixed radix filterbank that can be reused in future projects, such as SKA [7]. Second changing such a fundamental parameter requires all developers to revisit and verify their code again (VHDL, Python, C etc.) which typically will enhance the overall quality.

3 Technical impact

3.1 N=800 point FFT

3.1.1 FPGA resources

The polyphase filterbank (PFB) for B_{sub} = 781250 Hz uses a N=1024 point FFT [1]. In total the bn_filterbank design in the Apertif beamformer has two PFB to be able to filter S_{BN}=4 real ADC input signals per UniBoard back node (BN) FPGA. For B_{sub} = 1 MHz the FFT needs to be changed into a N=800 point FFT.

The Stratix IV FPGA that is used on UniBoard has 1288 multiplier DSP elements and 1235 M9K RAM blocks. Figure 1 shows that in total the bn_filterbank design uses 872 multiplier DSP elements and 747 M9K RAM blocks, so there are still maximum 1288 – 872 = 416 multiplier DSP elements and 1235 – 747 = 488 M9K RAM blocks available. The bn_filterbank contains two poly-phase filterbanks (PFB). The current N=1024 point FFT in the PFB takes 148 multiplier DSP elements and 39 M9K RAM blocks. Hence for the N=800 point FFT there are still 416 (free) / 2 (PFB) + 148 (current N=1024 FFT) = 356 multiplier DSP elements available and 488/2+39 = 283 M9K RAM blocks available. First impression is that this is more than sufficient.

Fitter Resource Utilization by Entity

	Compilation Hierarchy Node	ALMs	Combinational ALUTs	Block Memory Bits	M9Ks	M144Ks	DSP 18-bit Elems	DSP 9x9	DSP 18x18
1	|bn_filterbank	78864 (0)	80197 (1)	5400948	747	8	872	16	556
1	|ctrl_unb_common:u_ctrl|	3931 (2)	3545 (4)	121728	18	0	0	0	0
2	|node_bn_filterbank:u_node|	73421 (0)	74171 (0)	4184820	718	0	872	16	556
1	|bn_filterbank_ss:u_bn_filterbank_ss|	7087 (0)	5682 (0)	821708	124	0	0	0	0
2	|common_mem_mux:\gen_pfb_out:u_mem_mux_coefs|	677 (677)	680 (680)	0	0	0	0	0	0
3	|common_mem_mux:\gen_pfb_out:u_mem_mux_sst|	85 (85)	96 (96)	0	0	0	0	0	0
4	|node_bn_capture:\gen_pfb_out:u_node_bn_capture|	10957 (0)	12527 (0)	608768	84	0	48	16	16
5	|node_bn_terminal_bg:\gen_terminals:u_node_bn_terminal_bg|	31157 (0)	34204 (0)	1095680	132	0	0	0	0
6	|wpfb_unit:\gen_pfb_out:gen_wideband_poly_phase_filter:0:u_wide_ppf|	12634 (51)	10705 (0)	829332	189	0	412	0	270
7	|wpfb_unit:\gen_pfb_out:gen_wideband_poly_phase_filter:1:u_wide_ppf|	12428 (103)	10277 (0)	829332	189	0	412	0	270
1	|common_mem_mux:u_mem_mux_sst|	4 (4)	5 (5)	0	0	0	0	0	0
2	|fft_r2_wide:\gen_pfb:gen_wide_band_fft:gen_prep_wide_fft_streams:0:u_fft_wide|	7972 (0)	6371 (0)	182164	39	0	148	0	134
1	|common_requantize:\gen_output_requantizers:0:u_requantize_output_im|	9 (0)	17 (0)	0	0	0	0	0	0
2	|common_requantize:\gen_output_requantizers:0:u_requantize_output_re|	9 (0)	17 (0)	0	0	0	0	0	0
3	|common_requantize:\gen_output_requantizers:1:u_requantize_output_im|	9 (0)	17 (0)	0	0	0	0	0	0
4	|common_requantize:\gen_output_requantizers:1:u_requantize_output_re|	9 (0)	17 (0)	0	0	0	0	0	0
5	|common_requantize:\gen_output_requantizers:2:u_requantize_output_im|	9 (0)	17 (0)	0	0	0	0	0	0
6	|common_requantize:\gen_output_requantizers:2:u_requantize_output_re|	9 (0)	17 (0)	0	0	0	0	0	0
7	|common_requantize:\gen_output_requantizers:3:u_requantize_output_im|	9 (0)	17 (0)	0	0	0	0	0	0
8	|common_requantize:\gen_output_requantizers:3:u_requantize_output_re|	9 (0)	17 (0)	0	0	0	0	0	0
9	|fft_r2_par:\create_par_fft:u_fft|	937 (0)	360 (0)	0	0	0	16	0	12
10	|fft_r2_pipe:\gen_pipelined_ffts:0:u_pft|	1706 (0)	1496 (0)	18868	7	0	32	0	30
11	|fft_r2_pipe:\gen_pipelined_ffts:1:u_pft|	1499 (0)	1334 (0)	18432	4	0	36	0	32
12	|fft_r2_pipe:\gen_pipelined_ffts:2:u_pft|	1557 (0)	1472 (0)	52704	20	0	32	0	30
13	|fft_r2_pipe:\gen_pipelined_ffts:3:u_pft|	1504 (0)	1353 (0)	18432	0	0	32	0	30
14	|fft_sepa_wide:\gen_separate:use_wideband_separator:u_separator|	863 (24)	220 (40)	73728	8	0	0	0	0
3	|fft_wide_unit_control:\gen_pfb:u_fft_control|	1243 (409)	427 (82)	0	0	0	0	0	0
4	|fil_ppf_wide:\gen_pfb:u_filter|	2635 (0)	3149 (0)	524288	121	0	256	0	128
5	|st_sst:\gen_stats:gen_stats_streams:0:gen_stats_wb_factor:0:u_subband_stats|	192 (33)	109 (0)	30720	17	0	2	0	2
6	|st_sst:\gen_stats:gen_stats_streams:0:gen_stats_wb_factor:1:u_subband_stats|	160 (31)	72 (0)	30720	4	0	2	0	2
7	|st_sst:\gen_stats:gen_stats_streams:0:gen_stats_wb_factor:2:u_subband_stats|	162 (31)	72 (0)	30720	4	0	2	0	2
8	|st_sst:\gen_stats:gen_stats_streams:0:gen_stats_wb_factor:3:u_subband_stats|	150 (32)	72 (0)	30720	4	0	2	0	2
3	|sld_hub:auto_hub|	82 (52)	98 (65)	0	0	0	0	0	0
4	|sopc_bn_filterbank:u_sopc|	1907 (1)	2382 (1)	1094400	11	8	0	0	0

Figure 1 FPGA resource usage for the bn_filterbank design in the Apertif BF

Note that the implementation runs at 200 MHz, whereas the samples arrive at 800 MHz. This factor P=4 is called a wideband factor and causes that the implementation requires about P=4 times more FPGA resources than it would if it could run on 800 MHz. However the FPGA cannot run much faster than 200 MHz, so the factor P=4 cannot be reduced.

3.1.2 Radix-5 sections

For B_{sub} = 1 MHz the PFB will need an N=800 point FFT. An N=800 point FFT can be decomposed into 2 radix-5 stages and 5 radix-2 stages because 5^2*2^5=800. Figure 2 shows the radix-2 section and radix-5 section [8]. The crossing lines and the dot indicate the butterfly operation. For the radix-2 the butterfly only requires an add and a subtract, but for the radix-5 the butterfly also involves some multiplies. The arrow

indicates the twiddle factor, which involves the multiplication by the exponent term in the DFT. The twiddle factor is placed after the butterfly which indicates decimation-in-frequency (DIF).

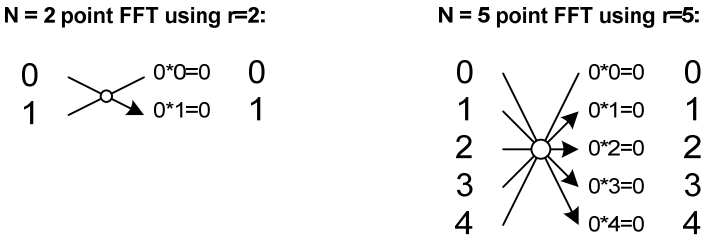

Figure 2: Radix-2 (left) and Radix-5 (right) decimation-in-frequency FFT

Figure 3 shows that a N=10 point FFT can be constructed as a radix-5 and radix-2 stage or with a radix-2 stage and a radix-5 stage. The stage order does influence the output order. The twiddle factors in the last stage have exponent 0 so these multiplications reduce to multiply by 1 and do not need to be implemented. In a pipelined FFT the input samples arrive in series. This implies that each stage only requires the hardware resources for a single butterfly + twiddle factor operation. The stages operate in parallel so the stages all do need their own resources.

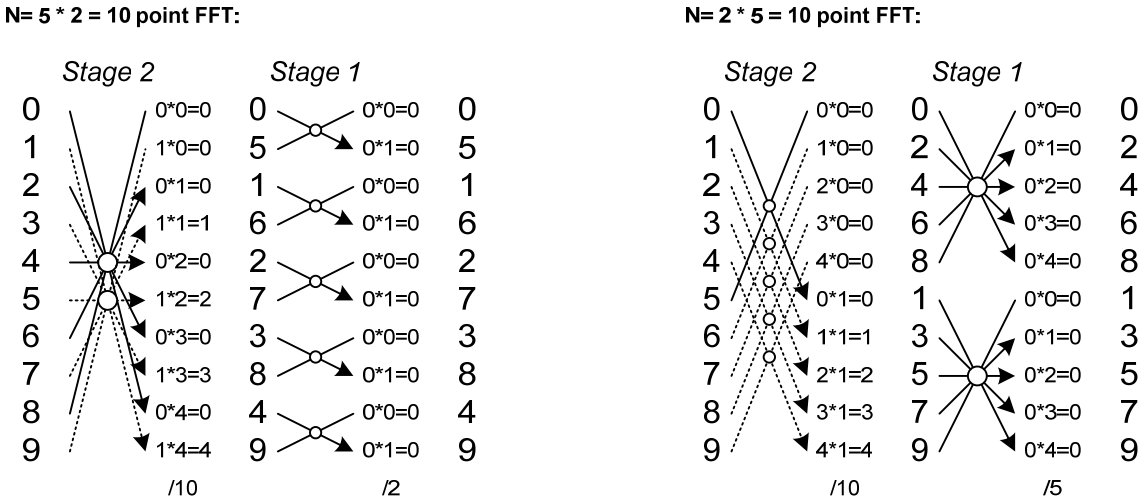

Figure 3: N=10 decimation-in-frequency point FFT using a radix-5 and a radix-2 stage

For example a N=1024 point FFT has $\log_2(N)=10$ stages, so it will need 10 complex multipliers in parallel to perform the twiddle factor multiplications. The multipliers operate at 50% efficiency because only 1 radix-2 butterfly output has a twiddle factor. Hence a radix-2 N-point FFT requires $0.5*N*\log_2(N)$ complex multiplies which agrees with the FFT theory [5]. An optimization is to avoid the multiply by 1 in the last stage. Another larger optimization is to try to use the multiplier at 100% instead of only 50% [7], but such optimization is not done in the current FFT implementation of the PFB in the Apertif BF [6]. Note that for a radix-r stage in general, the twiddle factor multiplier utilization is (r-1)/r, so for a radix-5 stage it is 4/5 = 80%, as indicated by the 4 arrows in the radix-5 butterfly in Figure 2. Hence trying to optimize the twiddle factor multiplier utilization in a mixed radix FFT becomes more difficult and less beneficial.

A N=800 point FFT has 2 radix-5 stages and 5 radix-2 stages. From a memory resource usage point of view it is better to use 2*2*2*2*2*5*5 than to use 5*5*2*2*2*2*2. However the parallel wideband factor P=4 implies that the last two stages need to b radix-2, so eg. 2*2*2*5*5*2*2. The FFT also need to reorder the output to normal order because the normal output order is necessary to be able to implement two real FFTs using one complex FFT. For an FFT with radix-5 stages the reorder is no longer a simple bit reversal.

Gianni Comoretto from INAF in Arcetri Italie has implemented a radix-5 FFT [7]. A synthesis trial for a N=3125 point FFT with 5 radix-5 stages reveals that stages 2, 3, 4, 5 each use 6 real DSP 18x18 multipliers that get mapped onto 8 DSP 18x18 elements. The memory usages increases for each stage and stage 4 uses 22 RAM M9K blocks.

Fitter Resource Utilization by Entity

	Compilation Hierarchy Node	ALMs	Dedicated Logic Registers	Block Memory Bits	M9Ks	M144Ks	DSP 18-bit Elements	DSP 18x18
1	|fft5s	3180 (1)	3915 (0)	926928	27	5	36	26
1	|fft5s_bf:i_stage0|	503 (0)	589 (0)	0	0	0	4	2
1	|fft5s_shuffle:i_reorder|	71 (71)	113 (113)	0	0	0	0	0
2	|winograd5_serial:i_winograd|	454 (454)	476 (476)	0	0	0	4	2
2	|fft5s_stage_n:\stage_g:2:i_stage|	666 (0)	834 (0)	4608	2	0	8	6
1	|common_async:i_dly_dav|	3 (3)	4 (4)	0	0	0	0	0
2	|common_async:i_dly_sop|	4 (4)	4 (4)	0	0	0	0	0
3	|fft_cm:i_cm|	18 (18)	36 (36)	0	0	0	4	4
4	|fft5s_bf:i_bfy|	507 (0)	589 (0)	0	0	0	4	2
5	|fft5s_shift_switch:p_input_buf|	48 (39)	62 (51)	2304	1	0	0	0
6	|fft5s_shift_switch:p_output_buf|	78 (68)	98 (87)	2304	1	0	0	0
7	|fft5s_twiddle:i_twiddle|	22 (22)	41 (41)	0	0	0	0	0
3	|fft5s_stage_n:\stage_g:3:i_stage|	631 (0)	812 (0)	23040	3	0	8	6
1	|common_async:i_dly_dav|	3 (3)	4 (4)	0	0	0	0	0
2	|common_async:i_dly_sop|	4 (4)	4 (4)	0	0	0	0	0
3	|fft_cm:i_cm|	18 (18)	36 (36)	0	0	0	4	4
4	|fft5s_bf:i_bfy|	499 (0)	589 (0)	0	0	0	4	2
5	|fft5s_shift_switch:p_input_buf|	45 (35)	68 (55)	9216	1	0	0	0
6	|fft5s_shift_switch:p_output_buf|	77 (64)	104 (91)	9216	1	0	0	0
7	|fft5s_twiddle:i_twiddle|	7 (7)	7 (7)	4608	1	0	0	0
4	|fft5s_stage_n:\stage_g:4:i_stage|	706 (0)	833 (0)	174456	22	0	8	6
1	|common_async:i_dly_dav|	3 (3)	4 (4)	0	0	0	0	0
2	|common_async:i_dly_sop|	4 (4)	4 (4)	0	0	0	0	0
3	|fft_cm:i_cm|	18 (18)	36 (36)	0	0	0	4	4
4	|fft5s_bf:i_bfy|	519 (0)	589 (0)	0	0	0	4	2
5	|fft5s_shift_switch:p_input_buf|	66 (56)	77 (61)	73728	9	0	0	0
6	|fft5s_shift_switch:p_output_buf|	101 (90)	113 (97)	73728	9	0	0	0
7	|fft5s_twiddle:i_twiddle|	10 (10)	10 (10)	27000	4	0	0	0
5	|fft5s_stage_n:\stage_g:5:i_stage|	693 (0)	847 (0)	724824	0	5	8	6
1	|common_async:i_dly_dav|	3 (3)	4 (4)	0	0	0	0	0
2	|common_async:i_dly_sop|	3 (3)	4 (4)	0	0	0	0	0
3	|fft_cm:i_cm|	18 (18)	36 (36)	0	0	0	4	4
4	|fft5s_bf:i_bfy|	509 (0)	589 (0)	0	0	0	4	2
5	|fft5s_shift_switch:p_input_buf|	69 (56)	83 (65)	294912	0	2	0	0
6	|fft5s_shift_switch:p_output_buf|	99 (84)	119 (101)	294912	0	2	0	0
7	|fft5s_twiddle:i_twiddle|	11 (11)	12 (12)	135000	0	1	0	0

Figure 4: Fitter resource usage report per stage for a N=3125 point radix-5 FFT

Assume that the resource usage of radix-5 stage 4 in the N=3125 point FFT is representative for the resource usage of the two radix-5 stages in the N=800 point FFT. Given the wideband factor P=4 this then implies that these two radix-5 stages will then require about 88 M9K RAM blocks and 32 multiplier DSP elements. This will easily fit given the resources that are still available in bn_filterbank design (see section 3.1.1).

3.2 Parameter change from N=1024 → N=800

The Apertif BF VHDL design is fully parameterized, so to changing N from 1024 → 800 can be done by simply changing the value. Still there will be impact at several locations, because there is no single central location where N is defined. The VHDL does use a central location, but the Python scripts and MAC software have their own central parameter locations. Therefore changing N will require rerunning several regression tests and fixing some unforeseen dependencies.

3.3 Processing and transporting N_{sel} = 304 subbands when B_{sub} = 1 MHz

The Apertif BF subband bandwidth is $B_{sub} = f_s/N$ = 800 MHz/1024 = 781250 Hz. The subband bandwidth can be increased to 1 MHz by using an N=800 point FFT in the subband filterbank. In total the Apertif must be able to process CB_{BW} = 300 MHz. Hence the number of subbands that need to be selected for the beamformer is N_{sel}= 384 for B_{sub} = 781250 Hz and N_{sel} = 304 for B_{sub} = 1 MHz, because N_{sel} needs to be

dividable by N_{band}=16 to be able to distribute the Aperif BF load over N_{band}=16 front node (FN) FPGAs in the Apertif BF subrack. The Apertif BF processing and data IO has sufficient spare capacity to handle CB_{BW} = 304 MHz, so it is not necessary to fall back to N_{sel}=288 which would fail the requirement of having CB_{BW} ≥ 300 MHz.

The descriptions below explain that processing CB_{BW} = 304 MHz is feasible. The description uses the load parameters for the Apertif BF that are defined in [3]. Figure 5 shows where the load parameters apply in the data path.

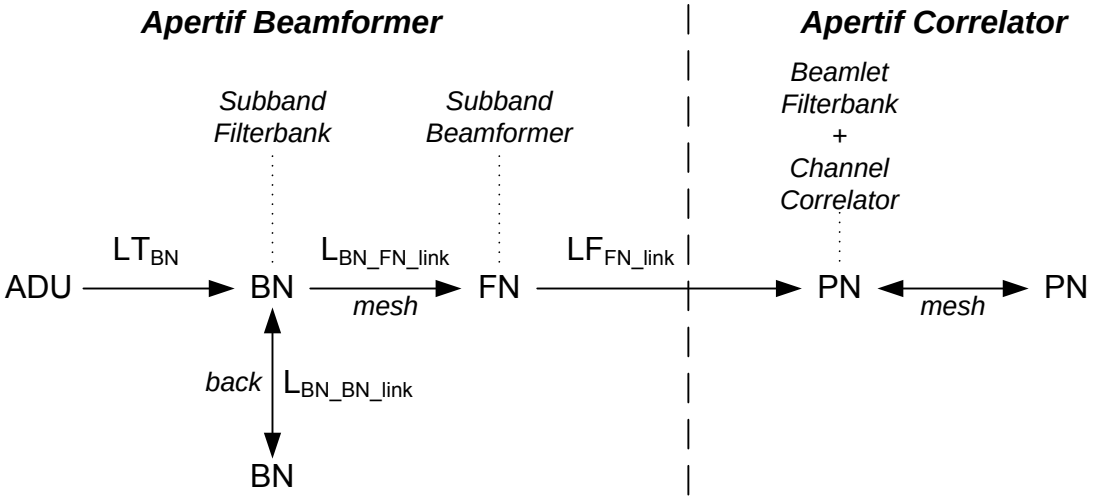

Figure 5 Load overview for the Apertif Beamformer

3.3.1 Processing load

The beamformer units in the FN can beamform N_{FN}=24 subbands (B_{sub}*N_{FN}= CB_{CW}/N_{band}= 300M/16 = 18.75 MHz) into P*N_{clk} = 4 * 256 = N = 1024 beamlets at the rate B_{sub}=781250 Hz. Hence the maximum number of beamlets per subband K_{max} = N / N_{FN} = 1024 / 24 = 42.6.

For the B_{sub}=1 MHz the same beamformer units will beamform N_{FN}=19 subbands (=B_{sub}*N_{FN}= 19 MHz) into P*N_{clk} = 4 * 200 = N=800 beamlets at rate B_{sub}=1MHz. Hence the maximum number of beamlets per subband K_{max} = N / N_{FN} = 800/ 19 = 42.1.

The processing load does not change, only the parameter settings of the beamformer units change and the balance between bandwidth CB_{BW} and number of beams K changes slightly. For B_{sub}=1 MHz the CB_{BW} increases from 300 to 304 MHz bandwidth and the maximum number of compound beams K decreases from 42.6 to 42.1, but is still more than the required K=N_{CB}=37.

3.3.2 Subband transport

Regarding the subband transport within the Apertif BF subrack the following points lead to the conclusion below.

1 In total the FPA of the Apertif BF has S=64 (actually 61, but the processing can handle 64) inputs per polarization that each get sampled by an ADC. The time series load per ADC signal path is LT_{sp} = 2 (Nyquist) * nof_adc_bits * RF_{BW} = 2 * 8 * 400M = 6.4 Gbps. There are S_{BN}=4 ADC signal paths per BN so the time series input load per BN is LT_{BN} = S_{BN} * LT_{sp} = 25.6 Gbps.

2 The subband filterbank outputs subbands with a data width of nof_pfb_bits = 16 bit (14b is sufficient, but assume 16b for easier transport using 32b words). The load per subband is $L_{subband}$ = 2 (complex) * nof_pfb_bits * B_{sub} = 2 * 16b * 781250 Hz = 25 MHz and will become $L_{subband}$ = 2 * 16b * 1 MHz = 32 MHz. The subband load per ADC signal path is L_{sp} = N_{sel} * $L_{subband}$ = 384 * 25M = 9.6 Gbps and will

become L_{sp} = 304 * 32M = 9.728 Gbps. The subband load per BN is $L_{BN} = S_{BN} * L_{sp}$ = 4 * 9.6G = 38.4 Gbps and will become L_{BN} = 4 * 9.728G = 38.912 Gbps.

3 The Apertif BF subrack has nof_uni=4 UniBoards. The load on the backplane in the Apertif BF subrack BN-BN links is $L_{BN\_BN\_link} = L_{BN}$ / nof_uni = 9.6 Gbps and will become $L_{BN\_BN\_link}$ = 9.728 Gbps. Each UniBoard has nof_fn=4 FN. Therefore the load on the UniBoard mesh BN-FN links is $L_{BN\_FN\_link} = L_{BN}$ / nof_fn = 9.6 Gbps and will become $L_{BN\_FN\_link}$ = 9.728 Gbps.

4 The capacity of the backplane links is 4 transceivers each operating at 5 Gbps including 8b/10b encoding, so an effective user data rate of 16 Gbps. The capacity of the mesh is 3 transceivers each operating at 5 Gbps including 8b/10b encoding, so an effective user data rate of 12 Gbps.

5 The subbands are transported in so called f-frames [3]. The frame rate is B_{sub} and the frame payload size $S_{BN} * N_{FN}$ where the number of subbands per FN is $N_{FN} = N_{sel} / N_{band}$ = 384 / 16 = 24 and will become N_{FN} = 304 / 16 = 19 (note that N_{band} = nof_uni * nof_fn = 4 * 4 =16). Hence the f-payload size is $S_{BN} * N_{FN}$ = 4 * 24 = 96 subbands and will become 4 * 19 = 76 subbands. The f-frame data width is 32b so given nof_pfb_bits=16b there fits 1 complex subband sample per frame data word. The f-frame overhead consists of 4 words for the Uthernet header [9] and tail and 4 words for the data path (DP) header and tail [10]. Hence the f-frame length is 8 + 96 = 104 words and will become 8 + 76 = 84 words. With a f-frame overhead factor of 84 / 76 the data rate for both the back plane BN-BN links and the mesh BN-FN links is (104 / 96) * 9.6 Gbps = 10.4 Gbps and will become (84 / 76) * 9.728 Gbps = 10.752 Gbps.

Conclusion:
The f-frame load for B_{sub} = 781250 Hz is 10.4 Gbps and for B_{sub}= 1 MHz it becomes 10.752 Gbps, but this still easily fits on the BN-BN backplane links (that can carry 16 Gbps) and on the BN-FN mesh links (that can carry 12 Gbps).

3.3.3 Beamlet output

Regarding the beamlet output of the Apertif BF subrack the following points lead to the conclusion below.

1 The subband beamformer in the FN can beamform N iblets. An iblet is a container that at the input of the beamformer carries a subband and at the output carries a beamlet. A beamlet can be regarded as a subband with direction. The Apertif BF must be able to produce K=37 beams, so on average N_{CB}=K=37 beamlets per subband. Each FN can maximum produce $K_{max} = N/N_{FN}$ = 1024 / 24 = 42.6 beams for B_{sub}= 781250 Hz and $K_{max} = N/N_{FN}$ = 800 / 19 = 42.1 beams for B_{sub}= 1 MHz.

2 The Apertif BF outputs beamlets that are $W_{beamlet}$ = 6 bits. The load for 1 beamlet is $LB_{beamlet}$ = 2 (complex) * $W_{beamlet}$ * B_{sub} = 9.375 Mbps and will become $LB_{beamlet}$ = 2 * 6b * 1 MHz = 12 Mbps. The beamlet output load per FN is $LB_{FN\_link} = N * LB_{beamlet}$ = 1024 * 9.375M = 9.6 Gbps and will be the same for B_{sub}= 1 MHz because then LB_{FN_link} = 800 * 12M = 9.6 Gbps.

3 The beamlet output load per FN is carried via a single 10GbE link. This link has a frame overhead of 12 (interpacket gap) + 14 (Ethernet) + 20 (IP) + 8 (UDP) + 14 (Data Path) = 68 octets. A block of N=1024 beamlets is packed into 64b words and then takes 2 * 6b * 1024 / 8b (octet) = 1536 octets for B_{sub} = 781250 Hz and 2 * 6b * 800 / 8b (octet) = 1200 octets for B_{sub} = 1 MHz. With 1 block per packet the Ethernet packet rate is B_{sub}.

4 With 1 block of N=1024 beamlets per Ethernet packet the required data rate is ((1536 + 68)/1536) * 9.6 Gbps = 10.025 Gbps and will become ((1200 + 68)/1200) * 9.6 Gbps = 10.144 Gbps. This just does not fit, so therefore the number of blocks per Ethernet packet needs to be set > 1. The 10GbE link can use jumbo frames up to about 8192 octets, hence it is possible to transport > 1 blocks of N=1024 beamlets per single Ethernet packet. Choosing N_{block} = 2 yields ((1536*N_{block} + 68)/(1536*N_{block})) * 9.6 Gbps = 9.8125 Gbps for B_{sub}=782150 Hz and ((1200*N_{block} + 68)/(1200*N_{block})) * 9.6 Gbps = 9.872 Gbps for B_{sub}=1 MHz, which fit 10GbE in both cases.

5 The frequency tolerance of 10GbE is 100ppm, so worst case the link can carry 9.999 Gbps instead of 10 Gbps.

Conclusion:
The Apertif BF output rate does not change when B_{sub} is increased to 1 MHz, because the number of beamlets decreases as much. The fact that the produced CB_{BW} increases from 300 MHz to 304 MHz does mean that slightly less than the original K_{max}=42.6 beams can be formed, but the number of beams that can be produced is K_{max}=42.1 which is still more than the required N_{CB} = K = 37. The number of beamlet blocks per Ethernet frame does not change either and needs to be at least N_{block}=2 for both B_{sub}=781250 and B_{sub}=1 MHz.

3.4 Channel bandwidth in the Apertif X

The channel filterbank in the Apertif X separates the beamlets into N_{chan} = 64 channels. For B_{sub}=781250 the B_{chan} = 12.207 kHz and for B_{sub} = 1 MHz this becomes B_{chan}=15.625 kHz. Having B_{chan}=15.625 kHz still fits the Apertif X requirement SYS-09-04 that the spectral resolution should be at least 20 kHz [2].