National Centre for Radio Astrophysics

Internal Technical Report
GMRT/egmrt/11-May-2018

FPGA-based Narrowband FPA Beamformer

EGMRT Internal Technical Report

Authors: *Atul Ghalame,*
Kaushal Buch, Ajith Kumar B

Emails: atulg@gmrt.ncra.tifr.res.in,
kdbuch@gmrt.ncra.tifr.res.in, ajit@ncra.tifr.res.in

Objective: Documentation of work done on the development of FPGA-based FPA beamformer. This ITR mainly covers 64-channel ADC based Narrowband Beamformer system for the focal plane array (FPA); including design, development and testing.

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

Chapter 1. Introduction

This document provides a detailed report of narrowband beamformer being developed for focal plane array feed elements. It consists first level signal processing after replacing single pixel feed with multi-pixel feed. Considering the number of feed elements to process, 64-channel ADC has been chosen for prototype narrowband beamformer development. An independent set-up has been prepared to interface the ADC to the FPGA board. A server has been configured with required utilities. A packetizer design has been developed to capture data from 64 channels with ADC-FPGA so that data can be processed by software. FPGA based beamformer design is being developed as a prototype to understand the working of Focal Plane Array beamforming. As the LOFAR system has a provision of recording data port from the RSP board, this option was also considered for the raw voltage recording.

Collaboration for Astronomy Signal Processing and Electronics Research (CASPER)[ref. 1] provides basic tutorials to use Reconfigurable Open Architecture Computing Hardware (ROACH) board. [ref. 1.a] The ROACH is a Virtex5-based upgrade of CASPER hardware. It works on a tool-flow environment prepared by integrating Xilinx and MATLAB-Simulink. Simulink design is simulated, compiled to generate bit file which programs ROACH (FPGA) boards [ref. 1.c]. These CASPER design tutorials have been used as the reference for subsequent narrowband beamformer development [ref. 1.b].

FPA beamformer background:

According to the eGMRT proposal, the field of view(FOV) would increase to 25 times that of single feed of the array. This will enable observation of a larger part of the sky. The requirement of FPA beamformer is to form multiple beams within the field of view pointing to different directions at a given point of time. All the feed elements are to be used to gain more sensitivity and narrower beam with aperture tapering. A central beam will be formed with all the feed elements receiving waves in phase. Phasing of feed elements is to be obtained by correlation and phase multiplication which would be the operation done after FFT. After phasing, the beam is to be steered to the nearby region by applying complex weights to the feed elements. In this way, we can have multiple beams in the enhanced field of view of the reflector dish. Amplitude scaling is needed for the feed elements to provide a tapering aperture, so that contribution from the elements can be controlled. Also, this can help to mark some of the feed inputs if they are not in operation or are not required in a beam.

Chapter 2. 64-channel ADC (64ADCx64-12)

The 64-input, 12-bit ADC board [ref. 1.d] s interfaced to ROACH-1 or ROACH-2 board by Z-DOK connectors. The board has 8 Texas Instruments ADS5272 chips, each digitizing 8 signals at 12-bit resolution and sampling rate up to 65Msps. The ADC can be clocked by an onboard oscillator or an external clock source. The 64ADCx64-12 is a twin Z-DOK card. Only one ADC board can be connected to a ROACH board. Fig.1 shows image of the 64-channel ADC board.

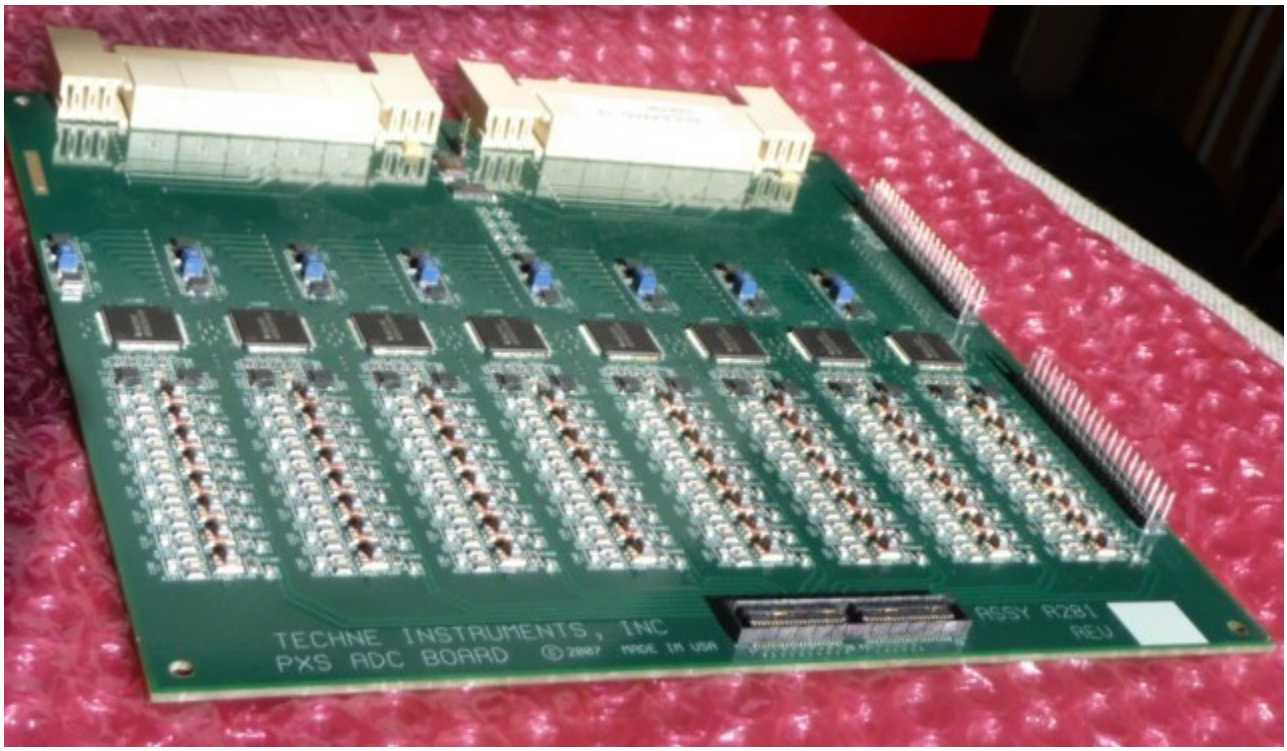


Fig.1 Image of 64-channel ADC

x64_adc CASPER block:

64ADCx64-12 [ref. 1.d] uses x64_adc yellow block interface (fig. 2) in CASPER environment. It interprets digitized data from 64 analog streams to FPGA in fixed point 12_11 format.

#Data input: Data can be provided for simulation using sim<n> and sim<sync> which is passed through as dout<n> and chan_sync.

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

#Data output: the 64-inputs are digitized by the ADC in 16 data output signals. Each signal will cycle through four multiplexed signals every four clock cycles. For example, in four consecutive clock cycles, a sample from channels 0,1,2,3 will appear on output "dout0". In the following four clock cycles, the next time sample will appear. Output dout<n> is responsible for samples from channels 4*n*, 4*n*+1, 4*n*+2 and 4*n*+3. Physically, ADC chip *m (m ranges from 0 to 7)* is responsible for channels 8*m*, 8*m*+1, ..., 8*m*+7. It is possible to identify the channels presented on each output by observing the chan_sync output, which is high when sample 4*n* is present on output dout<n>. The 8 bits of chan_sync are the sync flags associated with each of the 8 ADC chips. Proper calibration should ensure that all chips are synchronized. In synchronized case, the chan_sync output should output zero, with the value 255 appearing once every four clocks. [refer fig. 4]

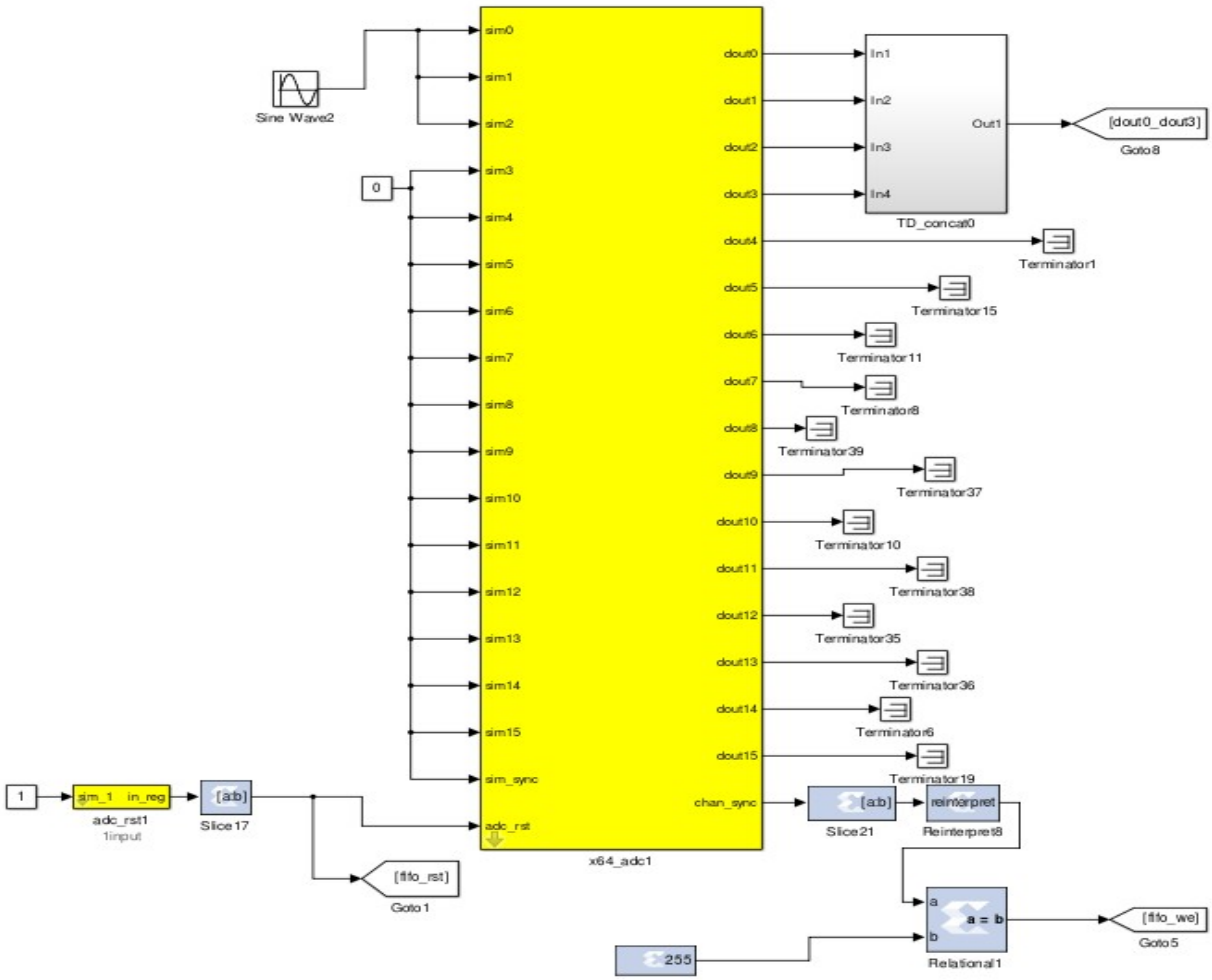


Fig.2 x64_adc CASPER yellow block interface

#Reset Signal:

The reset pin is located on pin 4 of J13. This signal is active low and should be held high for normal operation. The yellow block interface is configured to drive the ADC reset pin via GPIO<A|B>_0, depending on the block parameter specified by the user. Note that the reset on the yellow block interface is active HIGH. When the ADC is held in reset, data output on the yellow block data lines will have the value -1 for all channels, in fixed point 12_11 format. [ref. 1.d]

#ADC synchronization:

The ADC card uses 8 separate chips, each sending its clock over Z-DOK to the connector and FPGA. Rather than using all 8 clocks (some of which are not connected to clock enabled FPGA pins), a single clock is used, and the software calibration script is run to ensure that data from all ADC chips are properly aligned. [ref. 1.d]

Steps followed to interface ADC in GMRT:

i. ROACH-2 (Virtex-6 FPGA) was considered initially to interface 64-channel ADC.

ii. Basic design to acquire digitized data on 10GbE port complied and packets were acquired and observed on the Wireshark utility.

iii. Wake-On-LAN packets observed on Wireshark without providing any reset signal by a jumper on hardware.

iv. GPIO pin voltage of ROACH-2 was found to be 1.5V and was not sufficient to trigger ADC reset. So external voltage was given to reset, but ADC board started showing heating up of some of the ICs.

v. After suggestions from the CASPER community (https://casper.berkeley.edu/), a design for ROACH-1 was compiled and tested ADC interfaced to ROACH-1 board. ROACH-1 provides a 3.3V voltage which can trigger reset and has been used before by the CASPER collaboration.

vi. Finally, ADC data was observed on Wireshark as UDP packets by 64-channel ADC interfaced to ROACH-1. GPIO pin of ROACH-1 is used as a reset signal to the appropriate pin of ADC reset. Software register is used to create reset pulse for ADC.

Problems faced while interfacing ADC:

i. Interfacing to ROACH-2: 64-channel ADC interfaced to ROACH-2 has not been used much by CASPER community as per literature available online. As GPIO pin's high voltage is not enough to reset ADC board, ROACH-2 set needs some different arrangement for reset. We faced a problem of heating up of ADC ICs while interfacing to ROACH-2.

ii. ADC Reset: Reset pin 4 of the J13 header is active low; adc_reset of yellow block interface is active high which is connected to GPIO<A/B>_0 internally. GPIO<A/B>_0 has to be identified as per new mapping of ROACH board. The reset is controlled by software register connected to adc_reset of the yellow block in design. This reset pulse on GPIO_0 has been verified by a digital multimeter and found to be working when connected to ADC pin through a jumper. On successful reset WOL packets on Wireshark were no longer seen and replaced by UDP packets.

iii. XSG configuration: The XSG Core Config block is used to configure the System Generator design for the casper_xps tool-flow. Yellow block interface for x64_adc is designed such a way that when FPGA runs at clock rate 4 times faster than the x64_adc clock. When it is run with 50MHz onboard clock, FPGA would run at 200MHz clock. XSG_core_config has to be configured with User IP clock source as adc0_clk and user IP clock rate as 200MHz.

iv. Calibration script: There was a problem while running this script. While running the version available in the repository it threw the following error: unknown'x64_adc_ctrl' variable. This was fixed by adding the following line to the core_info.tab in the local repository:

location:

/mlib_devel/blob/master/xps_base/XPS_ROACH_base/core_info.tab

IF# strcmp(get(b,'type'),'xps_x64_adc')#x64_adc_ctrl 3 10000 100

Also, the following line of calibration script which is no longer supported by the 'corr' package can be removed (it seems this line was just meant for debugging).

#fclk_sampled = self.bit_string((val0&0x0fff),12)

Data integrity of ADC data:

Data integrity of digitized data from 64-channel ADC has been checked by a simple design which sends packets of data to a computer via 10GbE port. This data is captured by Gulp utility for few packets and unpacked by a script. Generated file of time domain signal is plotted back in GNUPLOT. Fig. 3 is a first basic result of data integrity of x64_adc:

- 1MHZ CW input -24dBm power applied to a channel
- Only one packet of 4K Bytes containing 4 serialized data shows 128 samples of an applied signal

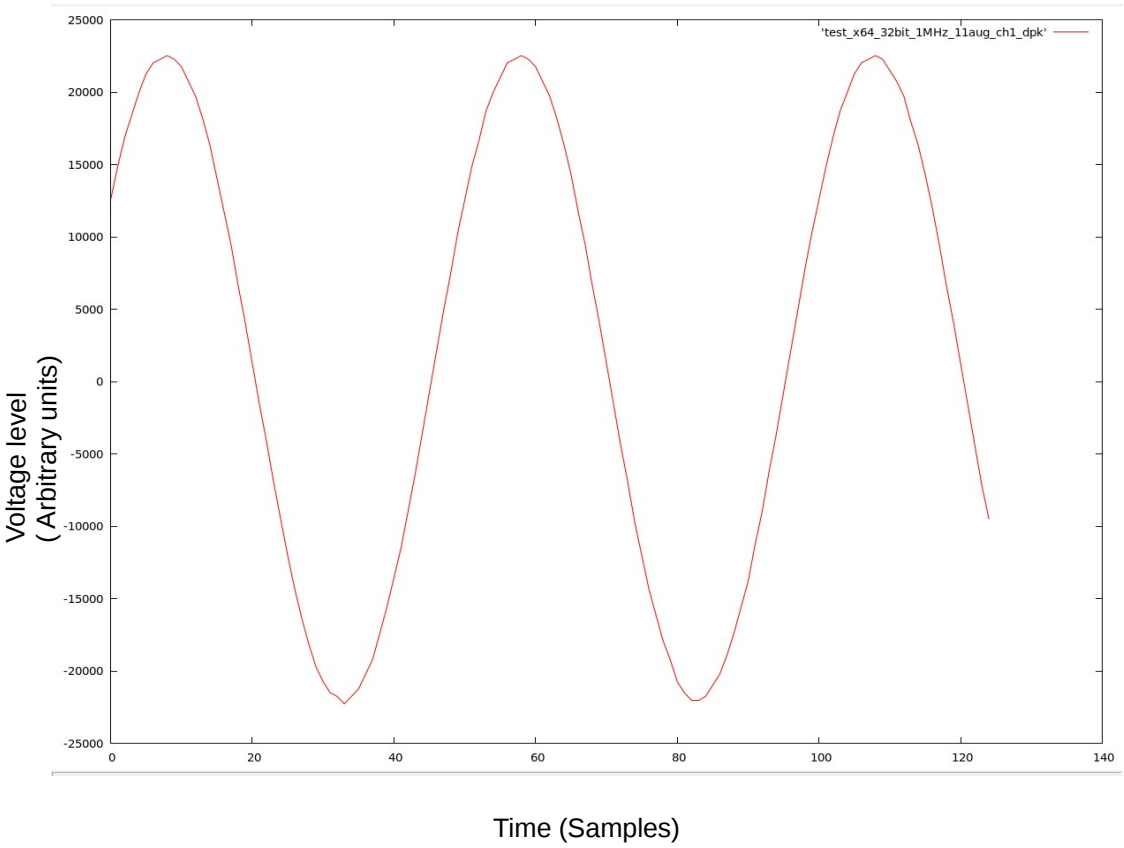


Fig.3 Data integrity test of x64_adc data of a channel in the time domain

Chapter 3. Narrowband Packetizer

A packetizer has been developed to acquire raw data from 64-channel ADC. ADC is interfaced to Virtex5 FPGA (ROACH-1) and data is stored into FIFO buffer, packeted, sent over 10GbE link to a computer. X64_adc accepts 64 analog RF inputs on a board by a high-density connector. After data integrity of digitized data has been tested, packetizer for all 64 inputs is needed for continuous data acquisition from ADC. If the sampling of the analog input is 50Msa/s, i. e. 25MHz bandwidth for Nyquist Sampling, hence the name narrowband Packetizer.

ROACH runs at 4 times faster to ADC at 200MHz, so the input data is much slower to send across 10GbE link. As explained in chapeter 2, 4 serialized inputs are provided to FPGA by one logical stream. There are 16 such streams carrying 64 inputs' data. So by the time one digitized sample of 4 inputs are present on physical pins, FPGA completes 4 clock cycles and can acquire data of 4 different inputs in each clock of ADC. This way of data acquisition has been used to store and packet slower data.

#ADC throughput: the data rate of x64_adc is:

e.g. 12bits X 64 inputs X 50MHz sampling = 38.4 Gigabits per second

This shows four 10 GbE ports would be required to send 12-bit precision data to a computer after digitization.

10GbE Interface:

As per the reference of CASPER community webpages [ref. 1.a, 1.b], ROACH boards have four CX-4 ports. There are two 156.25MHz crystals on the board. Each one clocks two ports, 0 & 1 and 2 & 3. This clock is then multiplied up on the FPGA by a factor of 20. Each port has 4 channels running in parallel (hence the digit in CX-4). Thus, the speed on the wire is actually 4 x 156.25MHz x 20 = 12.5Gbps. However, 10GbE uses 8/10 encoding, which means that for every byte sent, 10 bits are actually transmitted. For this reason, we actually get 12.5Gbps * 8/10 ~ 10Gbps usable data rate. CASPER's 10GbE Simulink core sends and receives UDP over IPv4 packets. These IP packets are wrapped in Ethernet frames with a header of 42 bytes. There is a ten_GbE_V2 yellow block from the CASPER XPS System Blockset to use transmission and reception of 10Gb data. In case of transmission reset, tx_data, tx_valid, tx_dest_ip, tx_dest_port, tx_end_of_frame are the inputs set in FPGA design.

-tx_valid has to be kept high when there is present to tx_data of the ten_GbE block

-tx_end_of_frame decides packet size, Max up to 8192 bytes in case of Jumbo packets.

-tx_dest_port and tx_dest_ip are set as per receiver side 10GbE card configuration.

Here, in case of x64_adc interfaced to ROACH-1, FPGA runs at 200 MHz clocks; this is faster than 10GbE core clock i. e. 156.25 MHz. Hence data from FPGA can't be sent over 10GbE core continuously. This has been done by setting tx_valid alternate high and leveraging the slower data rate of ADC output to pack frame in an optimum way.

#x64_adc channel identification :

As explained in chapter 2, 64 inputs' data is carried by 16 logical buses to FPGA by an interface named as dout0 to dout15 each carrying time multiplexed data of 4 inputs. Instantaneous channel on the data bus is identified by the chan_sync output of x64_adc interface. This goes high for 4N of inputs, n=0, 1, .. 15. Based on this channel are identified and segregated into different buffers. These can be classified as Group I, Group II, Group III, Group IV assuming Group I for 4N, Group II for 4N+1 and so on. Timing diagram is shown in fig.4.

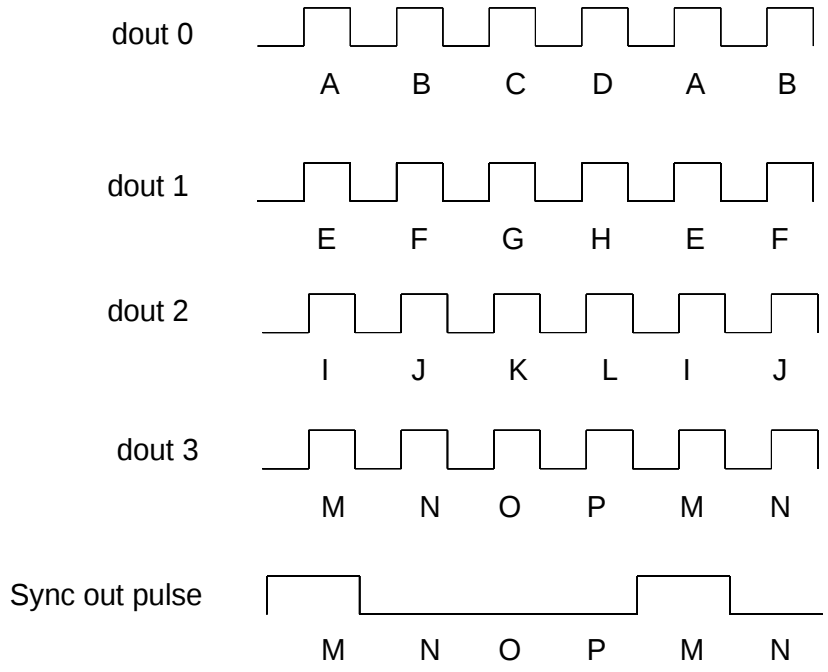
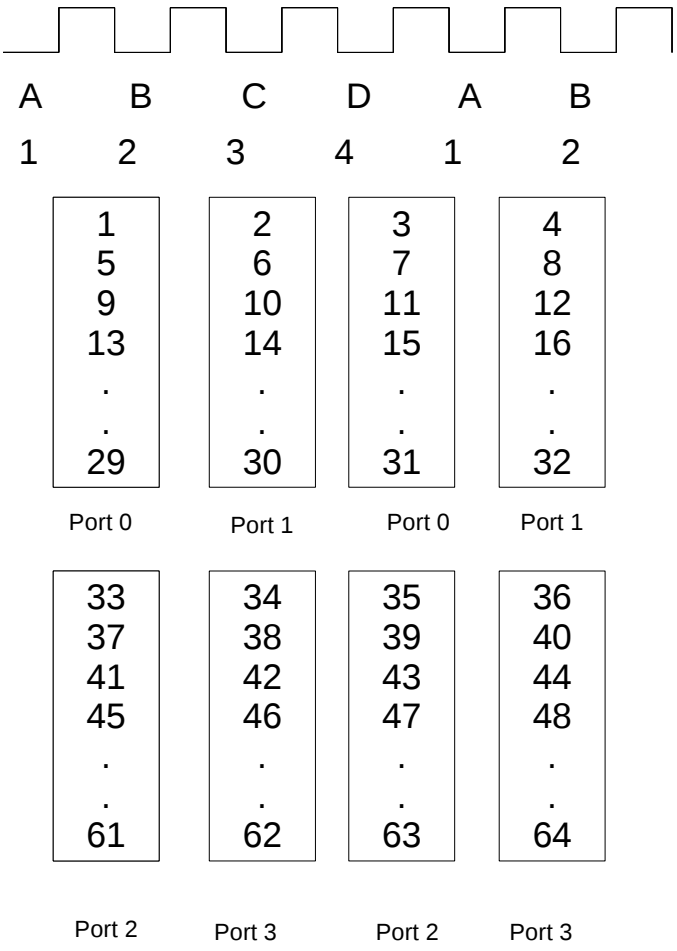


Fig.4 ADC Channel identification in FPGA design

The scheme used in Packetizer:

Packetizer grabs the data from all 64 inputs and sends frames to acquisition machine over 10GbE by four ports (fig. 5). This has been achieved in the best way ensuring no samples are lost and data flow is continuous. Data is divided into first 32 channels (dout0-dout7) and next 32 channels (dout8-dout15). x64_first_32 are stored in four FIFO buffers depending on the type of channel Groups. Then channels of Group I & Group III are sent via one first 10GbE port and Group II and Group IV are sent via a second 10GbE port. Same way is followed for next 32 channels of ADC. x64_next_32 data is given to 3rd & 4th 10GbE ports by segregating as G roup I, III and Group II, IV. As the data to the 10GbE core is sent in an alternate way, storing and sending alternate groups on one port makes sure that there is smooth data flow.



#Data format:

- Data word length: 64 bits
- 8 channels stored in one buffer form 1 word of data
- Next word consists of samples of other 8 channels

#Data rate:

- 64Msps sampling
- 64 digitized channels
- 8 bits per channel

==> 64M*64*8=32.77 Gbps
(Overall throughput)

#For 16 inputs per port

==> 8.192 Gsps

Fig. 5 Arrangemnet of 64 channels into 8 buffers sent over four 10Gb ports in packetizer

#Packetizer Specifications:

i. Packet size: 42 bytes of UDP header at the start of packet + 8 bytes packet count + 8192 bytes = 8242 bytes (payload 8200 bytes)

ii. Destination IP: 192.168.100.11

iii. Destination port: 60001

iv. 8 bytes of one value consists of 8 samples of different dout_n of interfaces. All 8 are from same Group and this Group repeats to every alternate data value on a 10GE link. Packet and channel details are explained in appendix-D

#The inclusion of PPS Sync: When a design is used in data pipeline or multiple boards are being used, PPS sync is used to trigger all the systems at a time. 64-channel ADC doesn't have a provision of external sync like that in i-ADC. This can be achieved by using SMA header on ROACH board and GPIO pins in design to get the pulse. The approach has been identified, test design is also ready but has not been tested on hardware. Also, one of the ADC input can be used as PPS input if ROACH header doesn't work.

Software processing for Packetizer:

Data from x64_adc will be packeted and sent to computing machine via 10Gb links. A continuous acquisition of data is underway to process the signal in the software domain. The code used in the 15m experiment is being tested to get the packets.

#Packetizer data on Wireshark:

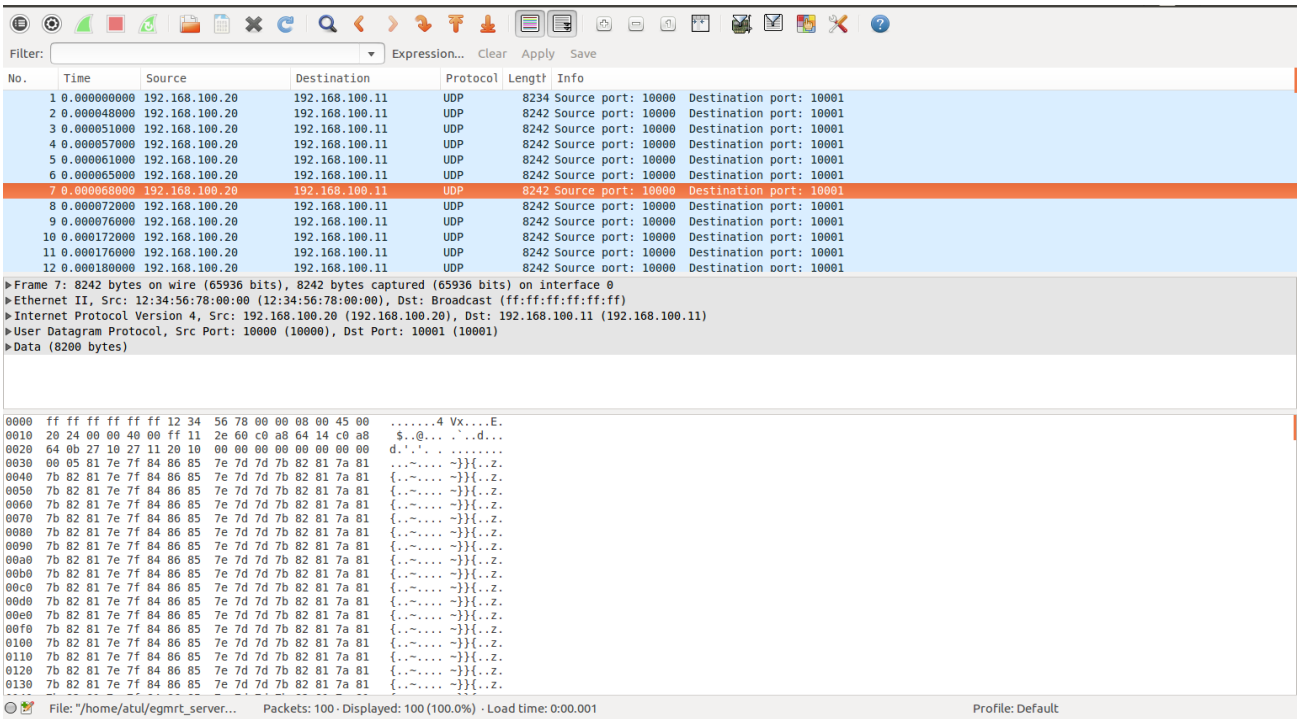


Fig.6 Screen-shot of data observed on Wireshark, UDP packets of 8242 bytes

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

#Two channels of ADC:

Data received from packetizer has been captured by Gulp utility and De-packeted by a code generating a file of time domain data. Following plot shows time domain signal of two channels of different ICs plotted at a time after calibration script is run. 3 packets containing signals of two channels have been plotted after calibration script is run. This shows both the signals seems almost in phase. 12-bit signed data has been reinterpreted to 8 bits.

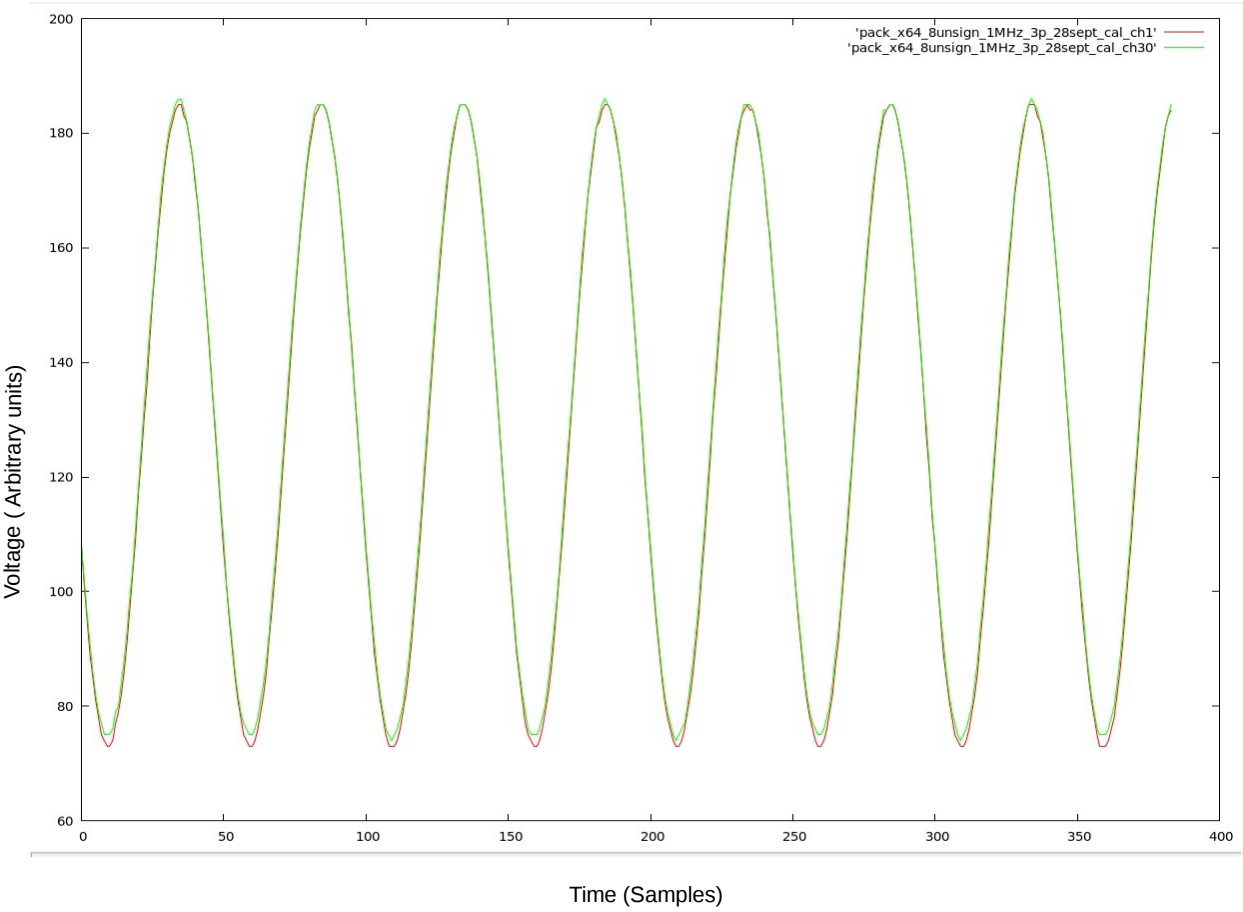


Fig.7 Data integrity test for two inputs after calibration

Chapter 4. 64-channel ADC based FPGA designs

Apart from software processing of 64-channel ADC based packetizer, the other option considered is FPGA based narrowband beamformer. Digitized data of ADC is identified and grouped properly and given to FFT blocks used in CASPER block set. With reference to CASPER basic tutorials (https://casper.berkeley.edu/wiki/Tutorials) step by step models for 64-channel ADC has been developed. CASPER tutorials [ref. 1.b] are prepared for wideband inputs to i-ADC. Understanding the working and general procedure, 64-channel ADC based spectrometer, incoherent beamformer, pocket correlation, coherent beamformer have been prepared. Each one of these case is explored with some basic results with few inputs. Once the signal flow and major problems are overcome, a final design for 64-channel, multiple beams, narrowband beamformer would be prepared.

General block diagram of x64_adc-FPGA designs:

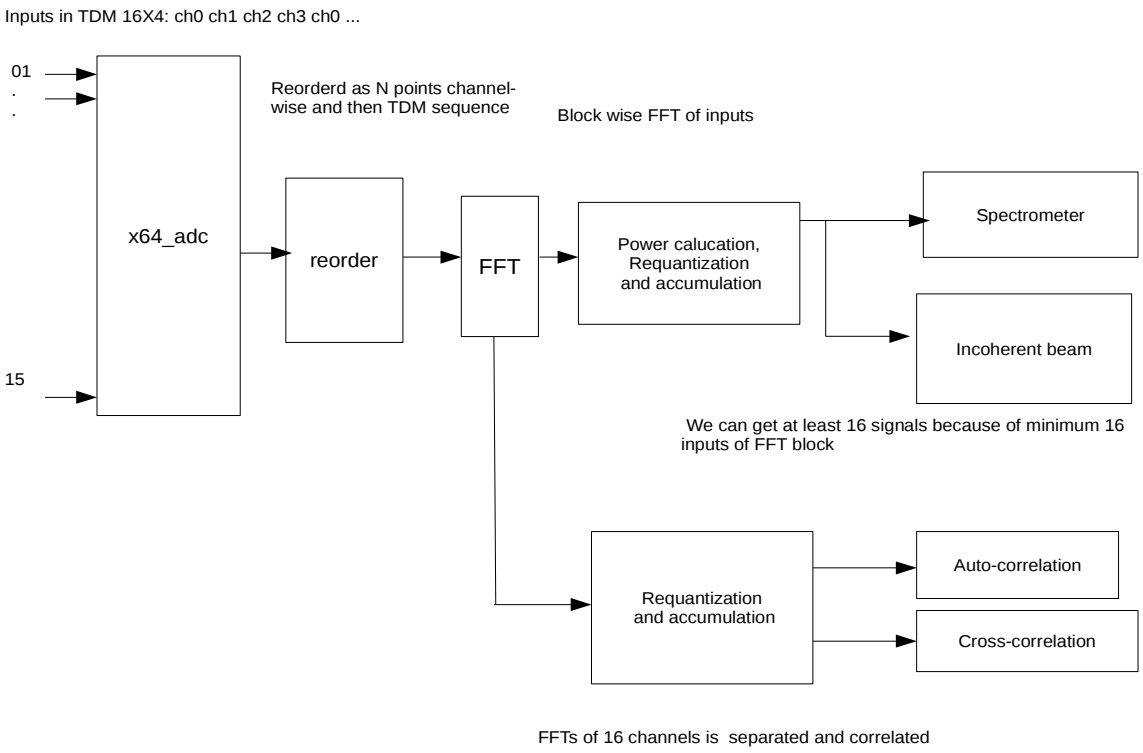


Fig.8 The overall block diagram of FPGA based designs

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

A. Spectrometer:

This is a basic step to proceed with FPGA based signal processing of ADC data. A spectrometer takes a signal in the time domain and converts it to the frequency domain. In digital systems, this is generally achieved by utilizing the FFT (Fast Fourier Transform) algorithm.

Bandwidth: BW of x64_adc based spectrometer is 25 MHz for onboard clock 50MHz

FFT channels: Different FFT points have been tried for an initial case study from 128 to 4096 FFT points. Only positive frequencies are considered, so FFT channels are half that of FFT points.

Frequency resolution:

$$\Delta f = \frac{BW}{no.\ channels}$$

e. g. for 256 FFT points, the frequency resolution is 25MHz/128= 195.312KHz
This resolution has been used for initial designs of correlation and beamforming.

Signal flow in design:

As depicted in general block diagram of 64-channel ADC based FPGA designs, digitized data from ADC obtained over 16 streams in FPGA need to be reordered. [fig. 9]. For example, dout0 stream gives signal a, b, c, d inputs one by one repeating after every 4 clock cycles. This mixed data can't be given to FFT as it comes. FFT in FPGA operates at 200MHz both input and output flow of data. Although there is serialized data, the slower rate of inflow can be used on reordering before FFT. 256-time domain samples of each a, b, c, d would need 1024 clock cycles of FPGA to gather the first input of each to FFT. Start of FFT can be delayed until the initial data is present to be fed to FFT. During 1024 clock cycles of FPGA, 256-time domain samples of each signal are gathered into different FIFO buffers. Method of channel identification is used to reorder the data. The chan_sync output of x64_adc yellow block interface identifies Groups(I-IV) of serialized signals and stores them into different FIFO buffers. After 1024 FPGA clock cycles FFT starts reading of first signal 'a' of dout0 till all 256 data points (N-points of FFT) followed by remaining b, c, d signals. This can calculate FFT of that particular block and give out FFT output again in a serialized manner. FFT_biplex_real_2x [ref. 1.e] has been used effectively in all FPGA based designs. Start of operation, reordering has been done by appropriate control signals.

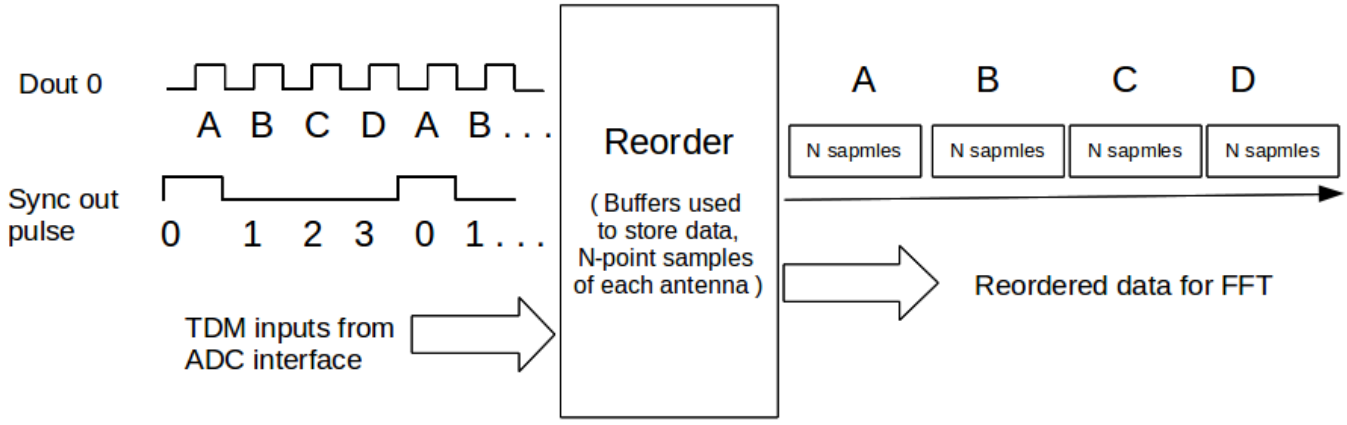


Fig.9 Reordering of ADC data for N-point FFT is considered

FFT_biplex_2x:

Computes the real-sampled Fast Fourier Transform using the standard Hermitian conjugation trick to use a complex core to transform two real streams. Thus, a biplex core (which can compute two complex FFTs) can transform 4 real streams. Twiddle-factor, and other logic sharing, allows multiples of 4 input streams to be processed simultaneously with minimal resource increment. Only positive frequencies are output (negative frequencies are the mirror images of their positive counterparts). Data is output in normal frequency order, meaning that channel 0 (corresponding to DC) is output first, followed by channel 1, on up to channel $2^{N-1}-1$. Real inputs 0 and 2 share one output port (with the data for 0 coming first, then the data for 2), likewise for inputs 1 and 3, and so on.

This CASPER block of FFT has been chosen as it doesn't require parallel inputs of time domain signals. i-ADC based CASPER tutorials use FFT_wideband_real which can't be used for x64_adc which gives data slower in rate and no parallel samples. Four samples from dout0 are reordered and sent to one input of FFT_biplex_2x block by block. If we apply a, b, c, d from dout0 of x64_adc to pol0_in as a first input to FFT and e, f, g, h from dout1 of x64_adc to pol3_in as a second input, we get FFT outputs on pol02_out of FFT_biplex_2x. This pol02_in gives out data clock by clock in the order of a, e, b, f, c, g, d, h. Only positive frequencies are given out from FFT, so we get 128

FFT output points of each signal one by one if N-point of FFT is 256. This means we need to observe all FFT data points coming out from pol02_out, i. e. 1024 frequency channel values containing 8 signals. In order to check the integrity of FFT data as a spectrum, 1024 values are plotted in subsequent design.

#FFT specifications:

i. Size of FFT: 256 points

ii. Input bit width: Fix_12_11 bits of ADC data is cast into Fix_18_11 number.

iii. Coefficient bit width: 18 bits used

iv. Shift to prevent overflow through FFT: 4096

v. Sync input is provided to FFT when data is available and sync out goes high when first FFT output is about to be given out.

vi. Quantization behavior is 'Round' type, Overflow behavior set to 'Saturate'

Spectrometer design flow after FFT:

The spectrometer is basically power spectrum. Power of the FFT output is calculated by squaring and adding complex FFT values. Calculated power is quantized to reduce the number of bits before it is accumulated into BRAM. The output of FFT_biplex_2x is 36bits for each FPGA clock cycle, 18 bits of the real part, 18 bits of the imaginary part both in fixed-point format. These complex numbers are separated into real & imaginary parts, squared and added properly calculating power in unsigned fixed (36 bits) format. These number bits are reduced by rounding off a number to 6 bits (fixed format) in quantization. This reduction provides larger time to accumulate data without saturation of value in BRAM. Vector accumulator block is used to store the data during each integration time. Length of vector accumulation is mentioned as block parameter and new accumulation pulse is also one of the input to set as per integration time. On every new_acc pulse, data is dumped in BRAM yellow block up to vector depth. The yellow block of BRAM is accessed by control machine through python scripts. In case of spectrometer 1024 FFT channels comprising 8 signals are repeating every time, so the vector length is set to 1024 in accumulator and BRAM addressing. All the design blocks are properly connected with delays and control signals. A separate logic works on generating accumulation pulse to synchronize the start of operation and accumulation of final data into BRAM.

#Experiment details:

- FFT points: 256
- Input: CW 15MHz, -27dBm

- Checked with varying channels among 4 TDM inputs
- All 12 ADC bits used (Fix_12_11)
- Re-quantized to Fix_6_5 after power calculation
- Integration time: 5.2 ms

Due to double serialization (in ADC & FFT), 8 signals are observed at a time in the spectrum. As mentioned earlier, we can get FFT as a, e, b, f, c, g, d, h when applied to the 0^{th} and 1^{st} input of FFT. Here in fig.10, only two signals viz. A & B were used and are visible as a peak. DC content of a, b, c, d is visible along with noise floor, but absent for signals e, f, g, h because zero input was applied directly to FFT as input. This was applied to test the order of spectrum data. Python script used in CASPER tutorial of wideband spectrometer [ref. 1.b] has been used with modification.

#Result of spectrometer:

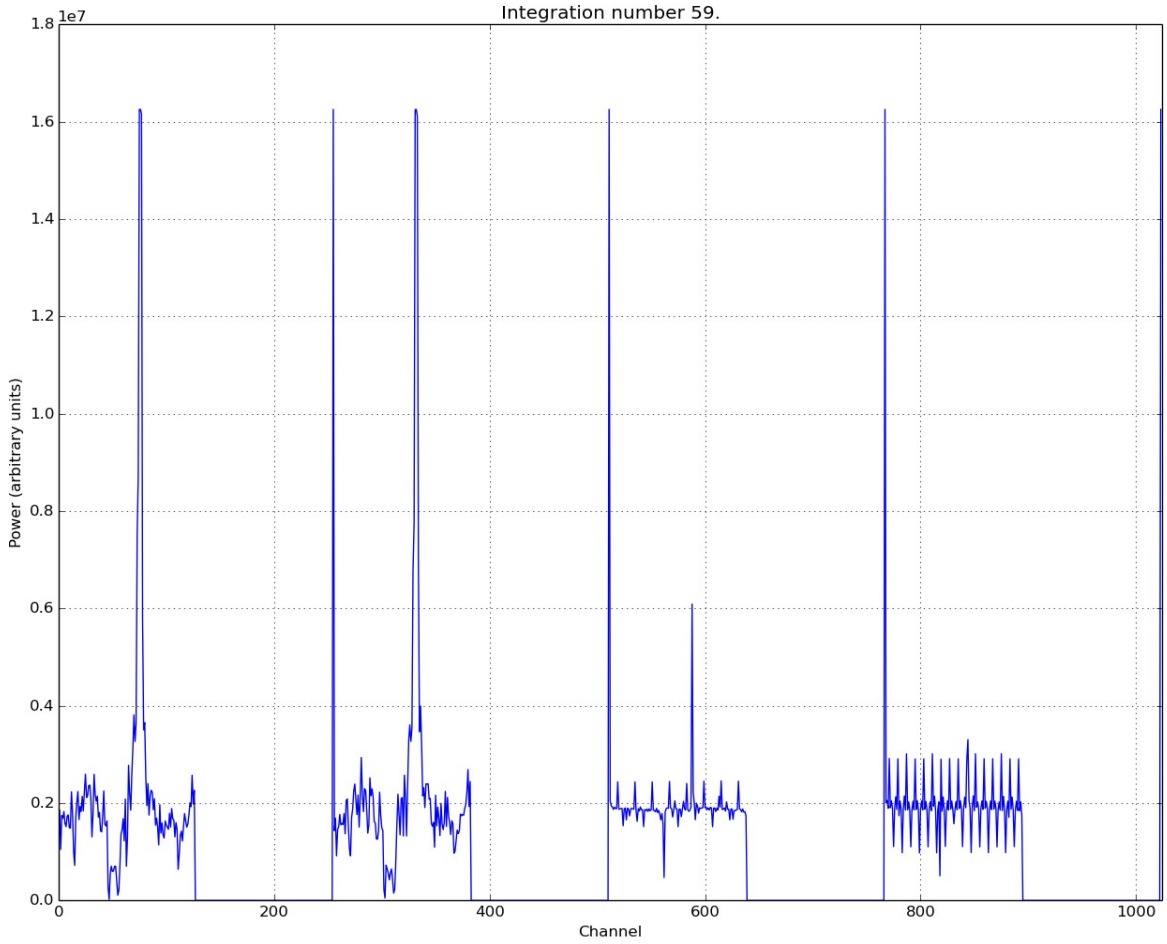


Fig.10 Eight serialized FFTs with tone frequency in two signals along with DC peak.

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

#Polyphase filter bank(PFB):

This block, combined with an FFT, implements a real Polyphase Filter Bank which uses longer windows of data to improve the shape of channels within a spectrum. Implementation of PFB has not been introduced because parallel streams of digitized data are not present in case of x64_adc. Data inflow and outflow has to be operated on 200MHz in this case, but ADC data is slower, so customized PFB has to be prepared to run 50MHz ADC data smoothly. The other option is to prepare control signals in such groups and samples that total number taps are applied, filtered and sent to FFT properly.

B. Incoherent beamformer:

On getting confirmation that a spectrum obtained digitally by CASPER design with different frequencies, an incoherent sum of power has been observed. In an incoherent beamformer, the power of each signal is calculated and added to other antenna powers. Phase coherence is not considered, just the amount of power from all antenna elements is added to get better signal strength. More the FFT points used, more the channels available and less is the noise floor in a power spectrum/incoherent addition. Incoherent beam:

Incoherent beam data = Power of the signal 1+ Power of the signal 2

$$V^2 = V_1^2 + V_2^2$$

There is little modification to spectrometer design where the power of a signal is already calculated and stored in BRAM. For an incoherent beam of x64_adc data, vector accumulator is more helpful as there are 8 serial signals coming out from FFT. Keeping vector length equal to the size of one FFT channels' instead of 8 makes the addition of powers of 8 signals. Complex values are cast to higher bits already to avoid overflow of data. This incoherent sum of powers along with spectrometer values has been kept in the same design in different BRAMs so as to debug results if required. Following fig.11 is one of the results observed using the incoherent addition of different signals.

#Experiment details:

- FFT points: 256
- Input: CW 15MHz, -27dBm
- One input applied only

- All 12 ADC bits used (Fix_12_11)
- Requantised to Fix_6_5 after power calculation
- Integration time: 5.2 ms

FFT_biplex_2x block used in all FPGA designs uses 4 inputs and gives out 2 outputs in a demultiplexed way. Already 4-time domain signals are present as ADC data on an FPGA bus, so we can get the Fourier transform of minimum 16 inputs at a time using same resources of CASPER blocks.

#Incoherent beam results:

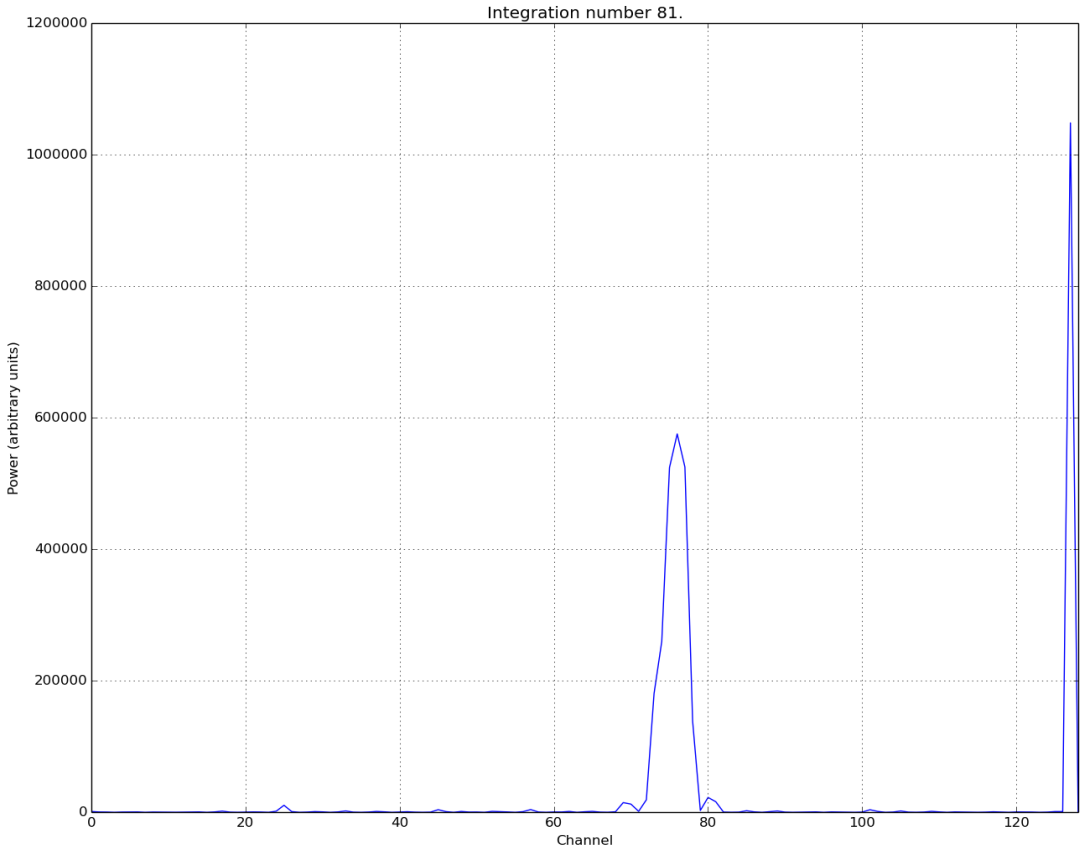


Fig.11 Incoherent beam two inputs applied

C. Designing a pocket correlator:

This section followed by the next section describes the early method used to explore the option of FPGA-based correlation and coherent beamformer. The pocket correlation means correlation done on a single ROACH board similar to CASPER-based to wideband pocket correlator. The same design of the wideband pocket correlator

was modified for 64-channel ADC data passed through FFT used before in spectrometer. The correlation subsystem requires continuous data for multipliers, so serialized of 8 inputs was given as input 1 and similar 8 inputs serially as input 2. The compiled design and corresponding scripts used in CASPER tutorials were found to be working and subsequently all the different pairs were tested. Plots shown in fig.12 is a cross-correlation of two such signals which are obtained by a power divider and dummy digital input in the design itself. Bigger peaks are actual CW signals along with DC peak nearby same as that of the spectrometer. Only two out of 8 serialized inputs were applied to both parallel streams of FFT output. Magnitude spectrum has power in arbitrary units and phase is in radian units in fig. 12.

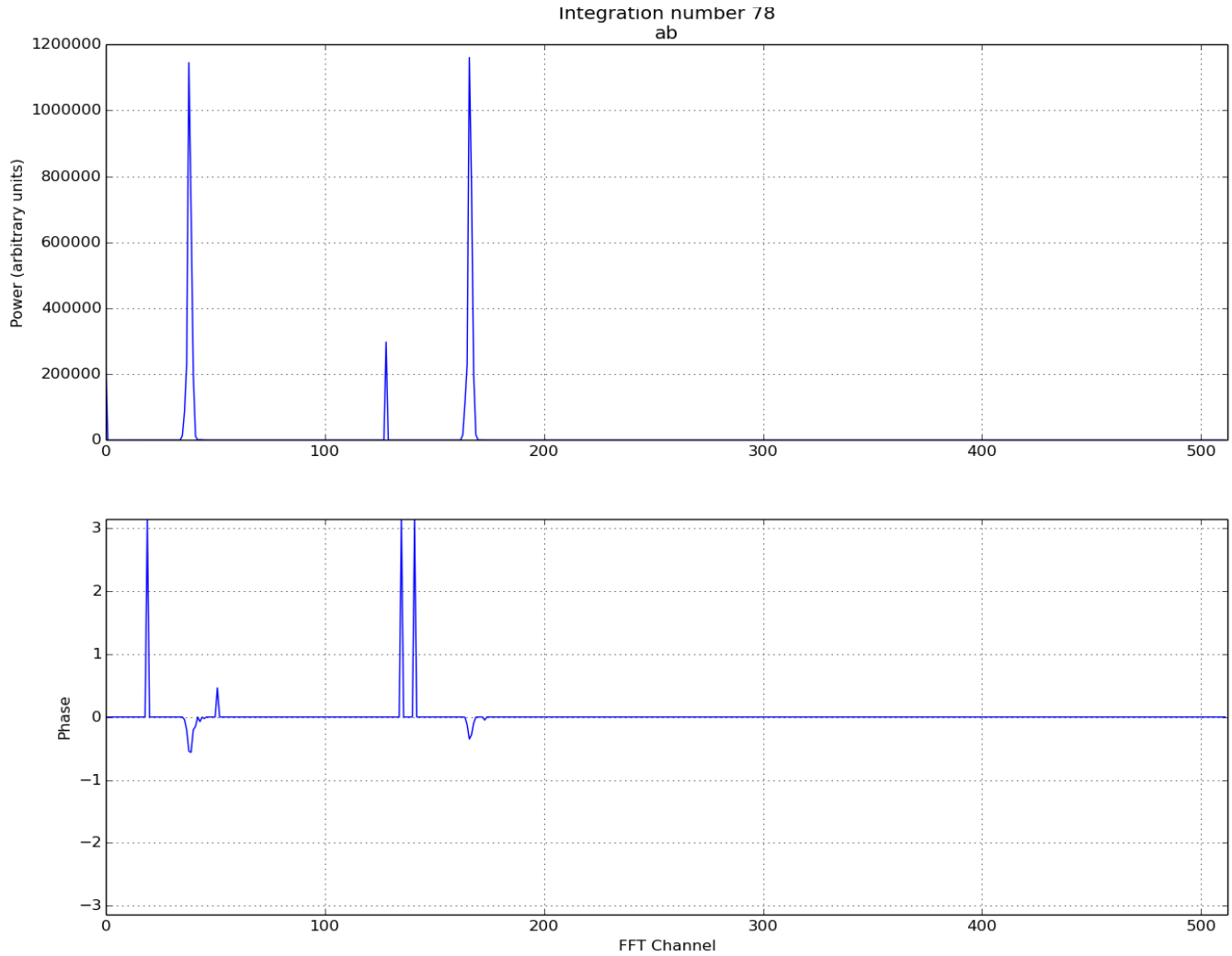


Fig.12 Cross-correlation of two signals made up of 8 serial inputs

Chapter 5. Development of correlator

All channels of ADC would be grouped in 8 numbers from FFT output, so correlation of different pairs can be calculated after separation of FFT data into respective buffers. Correlation has to be done in such a way that all signals arrive without any additional delay. Different methods have been tried to achieve proper correlation by trial and correction. Subsequent is one of the stable design which produced stable correlation magnitude and phase. Sync_out is a pulse given by FFT CASPER block which tells that FFT data would be available to next clock cycle. This initial pulse has been used in a precise way to identify the outputs coming out from FFT. Serialized FFT data is separated into 8 different buffers, which are read at a time to provide input to the correlator. Control logic resets counters, FIFO buffers on arrival of sync_out of FFT block. Control logic generates two signals one is FIFO_read and other is FIFO_write. The FIFO_write keeps on counting up to 8 which is compared with FIFO index and stored it matches. FIFO_write count is held constant for 128 clock cycles for 256 point FFT (half that of N points of FFT). FFT_read is enabled for 128 (N/2) clock cycles once all the FIFO buffers have concurrent data stored.

Quantization:

FFT_biplex_2x accepts 4 real-time sampled signals and produces Fourier transform of them in an alternate multiplexed way in two streams. Each FFT value is complex, 18 bits real, 18 bit imaginary both in Fix_18_17 format. These longer numbers are reduced by the quantization into Fix_4_3 format each. So the output of quantization block is an 8-bit unsigned number for each FFT value. This quantized value has been used to separate into FIFO buffers to save the bus width and resources.

Correlator:

Separated FFT data is carried forward to correlation. Two 8-bit unsigned numbers are spat into 4 bit real, 4 bit imaginary and multiplied by 'cmult_dsp48e' block of CASPER. This multiplies two complex numbers using 4 DSP48Es in a conjugate way. Vector accumulator is used for storing correlated output with the depth equal to the number of FFT channels. This accumulated data is stored into BRAM using a new_acc pulse of integration. Depth and addressing of BRAM are handled according to new_acc pulse. Data from BRAM is read by python script via 1Gb Ethernet connection of ROACH & PC.

#Initial Correlation experiments:

The above method was tried to get the correlation between two signals. All the design are based on simulation, trial and correction method. FFT data was needed to separate correctly, write into buffers, read at a time, and correlate. CW input has been used to check the working of correlation. CW input is expected to appear at the relevant channel of autocorrelation after splitting. While carrying experiments of correlation following erratic results were obtained:

#Experiment:

Inputs: CW, 15MHz signal applied by power divider.

FFT size: 256-point FFT, 128 FFT channels, Signal expected in the 77^{th} channel.

Correlation: the magnitude spectrum of cross-correlation should show only relevant peak and no phase between them.

In fig. 13, auto-correlation showed the perfection of splitting of signals by presence of a peak in 77^{th} FFT channel. Channel A is not exactly appeared but B, C, D are as expected.

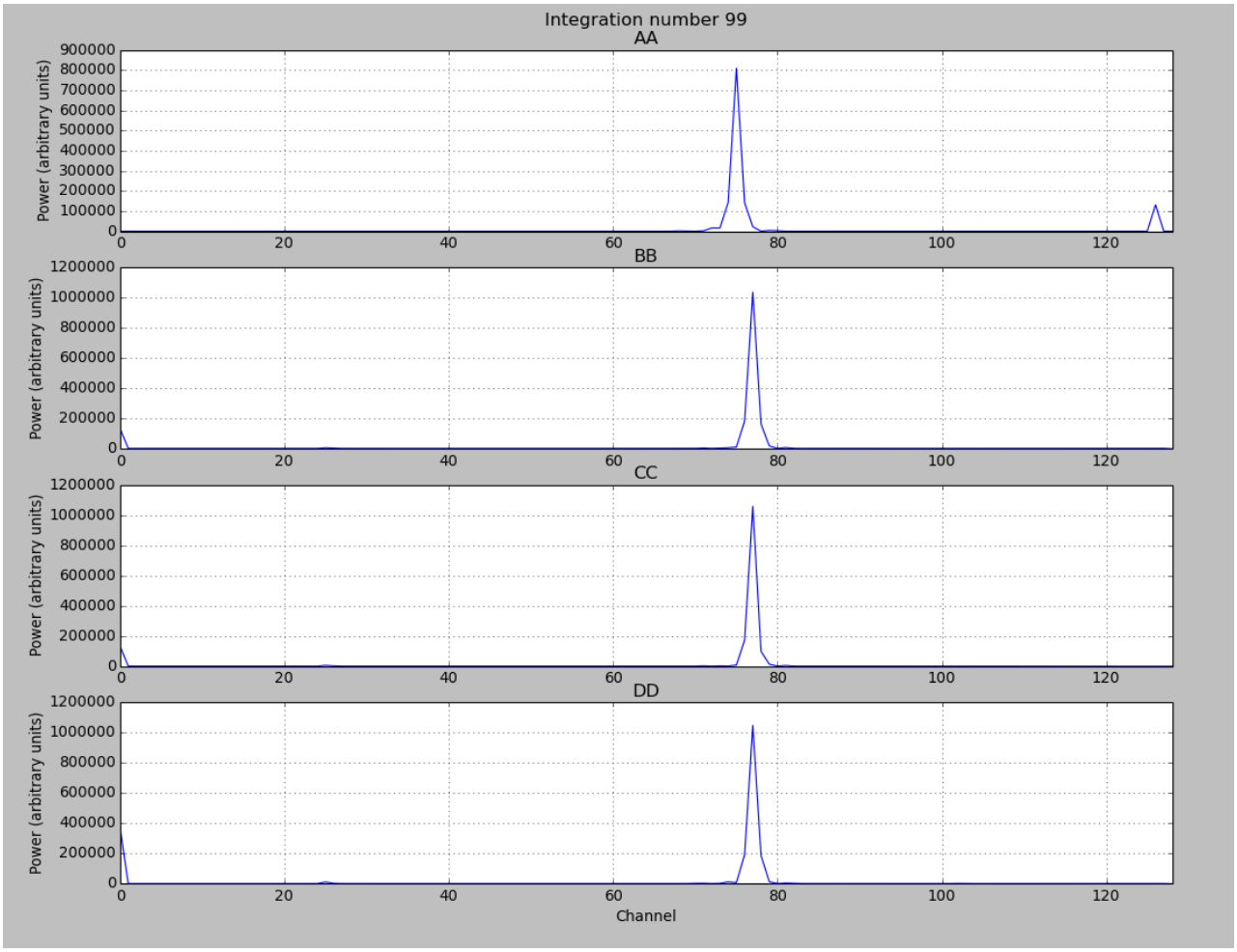


Fig.13 Auto-correlation of 4 signals after separation of FFT

As signal A is not showing proper auto-correlation of CW peak, only B & C were cross-correlated and the plot is shown in fig.14. It shows the magnitude of cross-correlation is right but phase isn't. The phase spectrum of correlation shows values nearby to CW peak and flat phase to other channels where there is no signal. Also, this phase spectrum was variable at runtime. The whole plot of the band goes up and down while running (fig.14). This was instability observed and needed design corrections explained later.

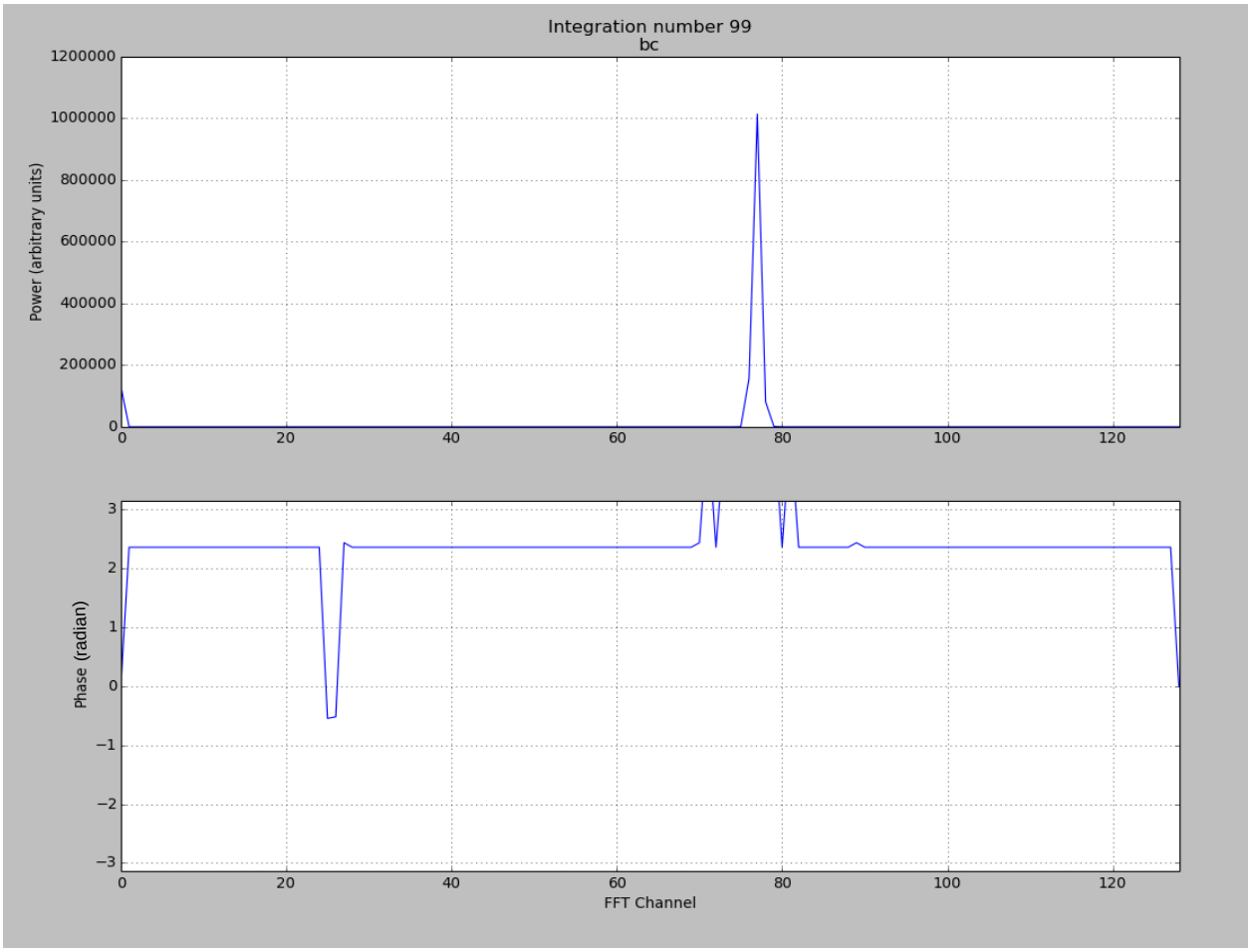


Fig.14 Unstable phase of cross-correlation

All the correlation plots are runtime observed by python scripts. The integration time is 0.6s set by accumulation control logic. This generates a pulse at every 0.6s so that data stored in all vector accumulator is stored into BRAM and read by the script.

Auto-correlation could be corrected by simulation & correcting read, write control signals of FIFOs but phase spectrum of cross-correlation didn't produce due to delay of a clock or two.

Cross-correlation stability:

While working on a stable correlation of signals, separation of serialized FFT data was ensured by prediction, simulations, and trial on hardware. This helped to get stable auto-correlation without any delay (fig.15) among 8 channels but phase was still problematic. The reason after debugging was found that inputs given to correlation are being given in burst mode, only when FIFO read is high, and this was high for 128 cycles out of 1024 clock cycles. This has been corrected in next designs by 'zero-padding' when there no data is read out from FIFO. This makes an arrangement so that correlator multiplier gets continuous data may that be both zero or concurrent data of both channels. This way has exactly made the correlation stable one. Following are the results of stable correlation and its testing. The same described above set-up has been used to test the stability of correlation for corrected designs.

Input: CW, 15MHz, applied through power divider,

FFT size: 256-point FFT, 128 FFT channels, Signal expected in the 77th channel.

Correlation: magnitude shows proper peak and phase is also absent and stable.

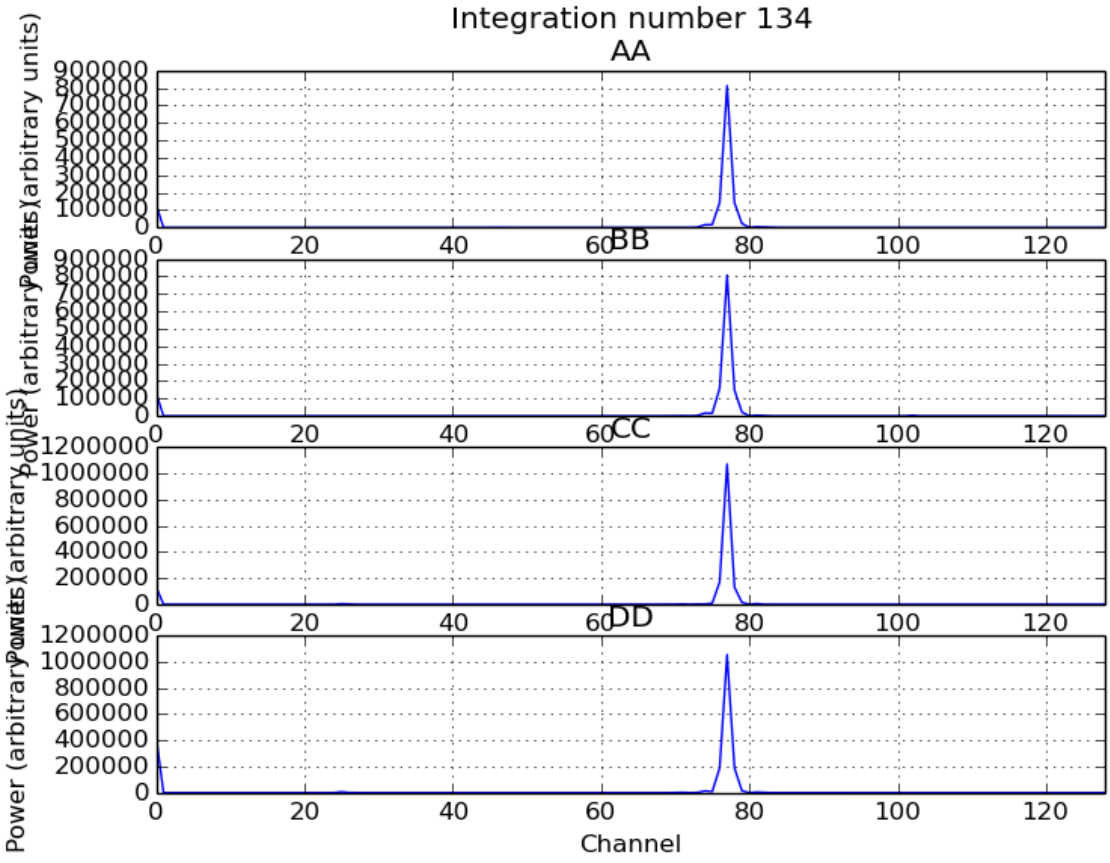


Fig.15 Stable auto-correlation of four signals without any delay

#Cross-correlation of A& B after separated FFTs:

- Input: CW 15Mz, -28dbm (same for this experiment till fig.15)
- FFT point: 256
- A signal is present in channel 77 as expected
- There is some phase between A& B as seen in the plot
- This is quite stable correlation than the earlier separation of FFTs
- Phase is plotted in degrees and magnitude in arbitrary units

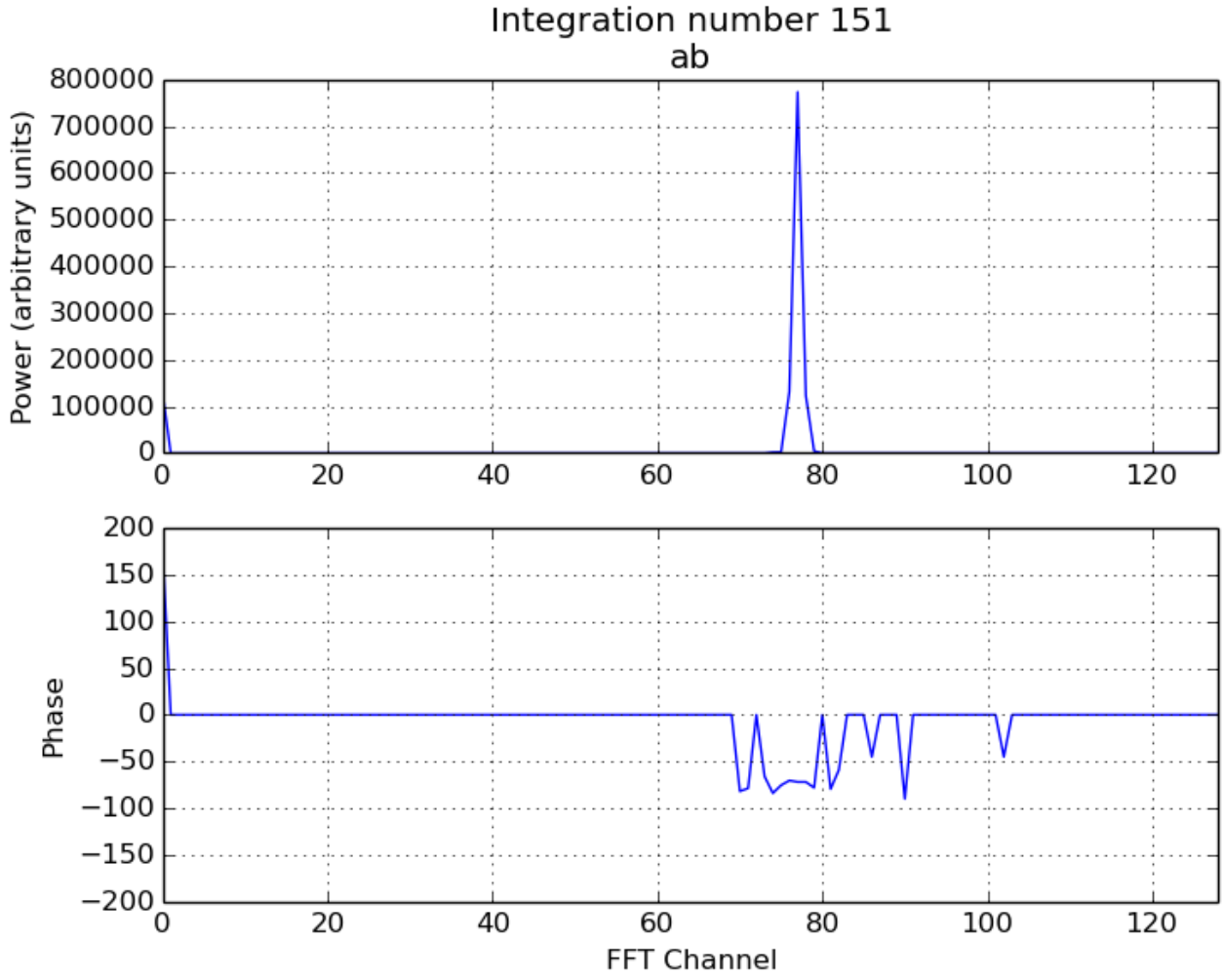


Fig.16 Magnitude & phase of cross-correlation of A & B

Magnitude shows exact CW peak but phase is present around CW, which is not expected but it is stable over time. This is solved as explained in appendix-C.

#Stability(of AB) check over time:*

- A single channel of cross-correlation has been plotted in the fig.17 plot over time.
- X-axis values are integration numbers; Y-axis values are arbitrary units & phase in degree similar to earlier plots. Initial shift is because 0^{th} channel was being plotted by default as design starts operating. Due to the presence of DC content, there is substantial magnitude and phase.
- When channel for stability is changed to 77 where the signal is present, magnitude goes high and the phase difference is reflected.
- There isn't data beyond X value 420 as the plot was saved while operating.
- Fig.17 shows the fair stability of cross-correlation between A & B

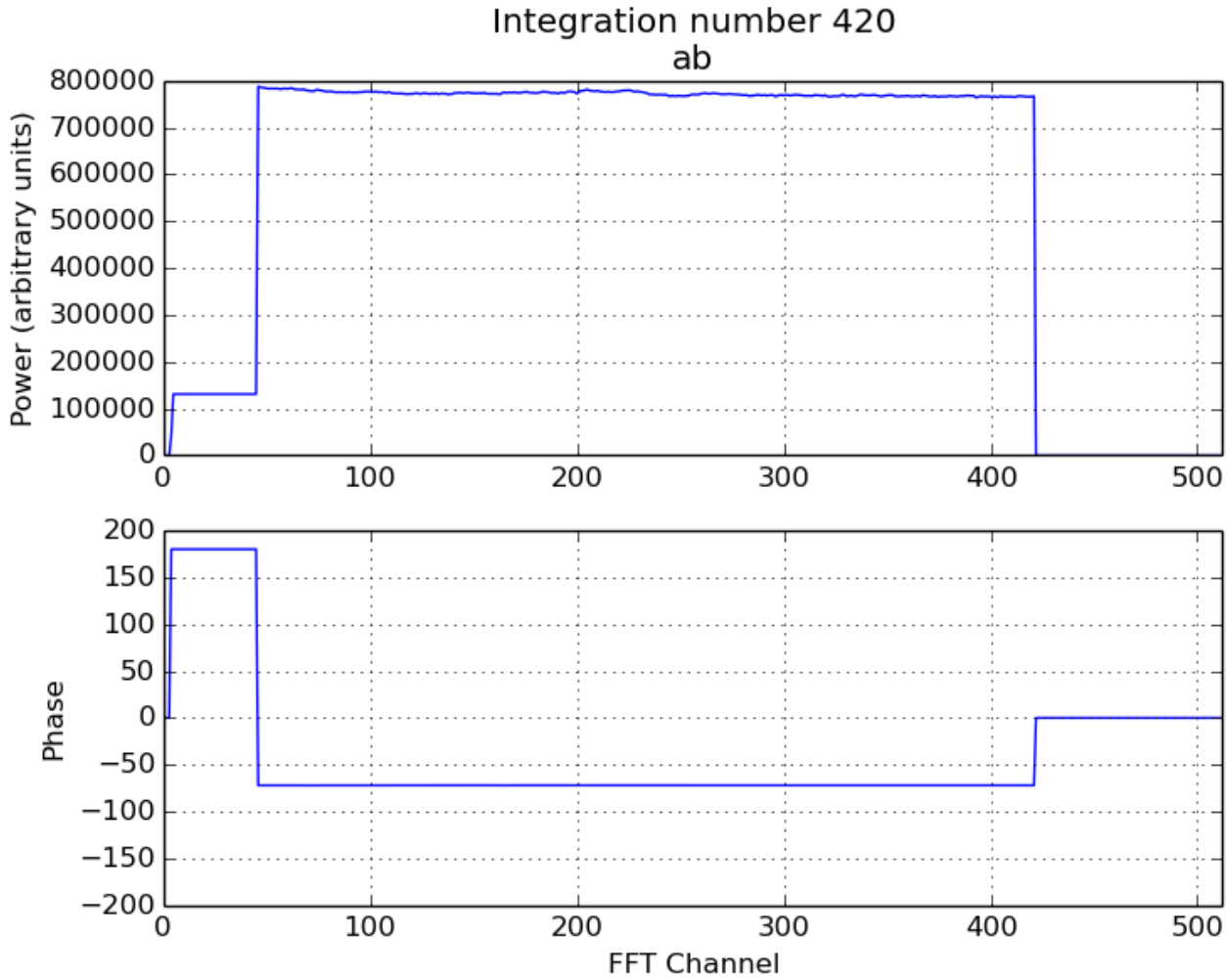


Fig.17 Stability cross-correlation of A & B check over time

The experiment was done with 4 signals(A, B, C, D) just. Out of which cross-correlation of A & B is shown in fig.17 as an example.

Chapter 6. Development of beamformer

As explained in earlier chapters, a digitized signal has been tested with data integrity, channel identification, Fourier Transform, correlation. The way to get stable correlation has been identified and discussed in appendix-C. Correlation provides phase between the signals which can be used to apply to one of them to make both signals in phase. This phase multiplication after FFT corrects non-integral delay between signals. This is chapter describes how phase is corrected and a beam is formed. Separated signals of serialized FFT output are correlated in pairs, real & imaginary parts of cross-correlation are stored in BRAM. The python scripts plots correlation, calculates phase values and stores in a file. Each pair of correlation provides a file of FFT channel wise phases. These phases of pairs have to be applied by multiplication to the corresponding signal after FFT despite serialized data.

Phasing of antenna signals:

Phase is stored and multiplied in terms of sine or cosine of obtained theta. It was found that it is simple to apply phases of all 8 serialized input at a time than a pair. All the phase values of 8 signals are stored in a list in python script to create a single list to be given through software register. CASPER block 'SineCosine' look-up table (LUT) is used to generate sine & cosine of theta value. Depth, output bit width, the delay is set for LUT as per requirement. Depth chosen is 2048 for better phase resolution. 360 degrees of phase is mapped to 2048 values. Phase value (theta) is calibrated in python script in accordance with the depth of LUT. Theta value is stored in single port RAM with addressing covering 8 serialized signals, for example, for 256 point FFT depth of RAM would be 1024. Sine & cosine of theta is multiplied exactly to FFT channel of corresponding antenna signal accordingly. One of the antennas from is considered as reference and all are correlated with that. There is a bit growth due to multiplication so data is truncated into (18 bit real + 18 bit imaginary) again (fig.18).

Amplitude scaling: As a first effort, 8 bits are used to multiply with phase multiplied value followed by truncation again. We can scale the antenna signal or flag its contribution to a beam by applying value through the script. Exact bits for scaling and truncation would be finalized later when to be tested with FPA in total (refer fig.18)

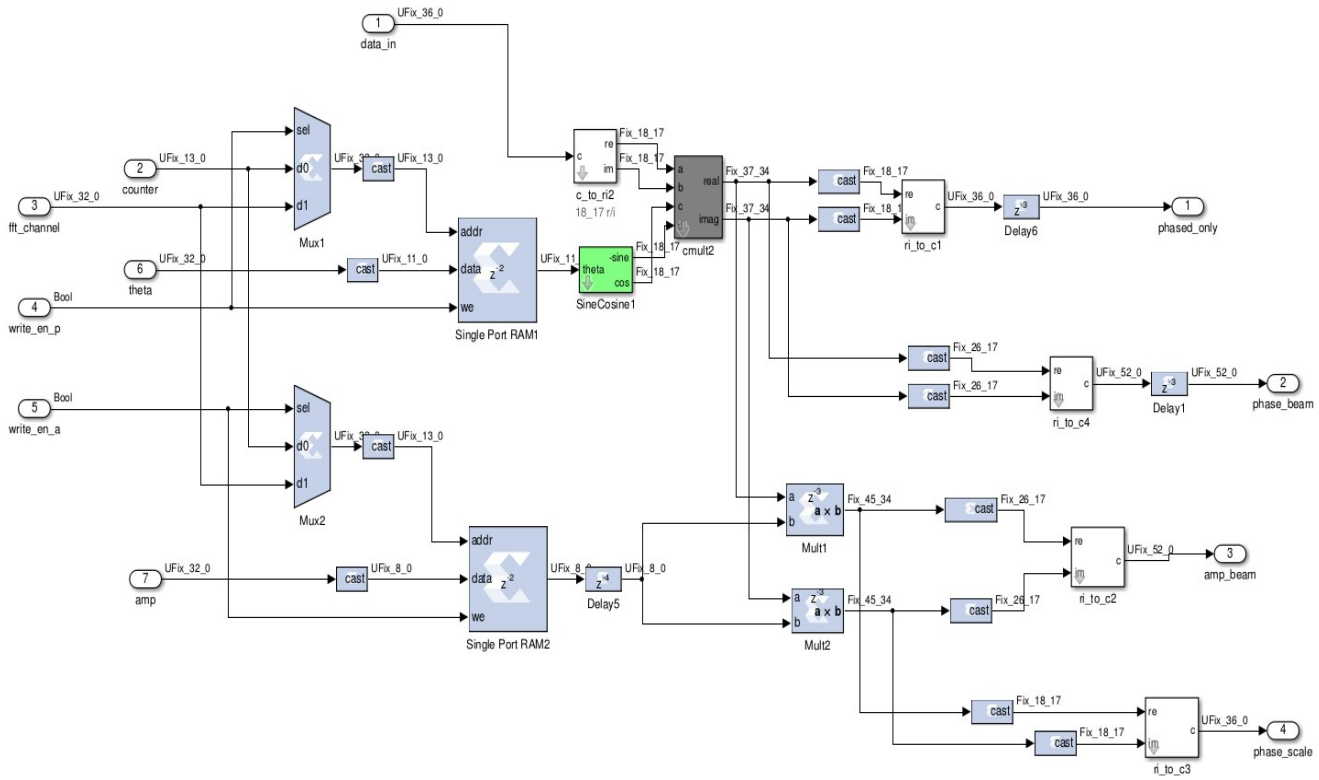


Fig.18 Weight multiplication for 8 serialized antenna signals of x64_adc

Coherent beam:

Phasing of antenna elements makes inputs coherent. Signal to noise ratio (SNR) increases N times in a coherent beamforming whereas it increases by root N times in an incoherent beam when N antenna signals are added. In a coherent beam, voltages of phased signals are added first and power is calculated.

For example, in case of 2 elements,

$$\text{Incoherent beam: } V^2 = V_1^2 + V_2^2$$

$$\text{Coherent beam: } V^2 = (V_1+V_2)^2 = V_1^2+V_2^2+2V_1V_2$$

Additional term of $2V_1V_2$ improves SNR compared to an incoherent beam.

Weight multiplication after FFT of antenna signal consists of both amplitude scaling of an antenna element and phase multiplication to correct phase and for beam steering. It is ensured that output of FFT and weight multiplication are of the same format. Weight multiplication gives two signals one for coherent beam and one for correlation. The signal before correlation undergoes quantization from 36 bit to 8bits (4 real, 4 imaginary). The signal forming coherent beam uses all 36 bits.

The FFT_biplex_2x block uses minimum 16 inputs to calculate Fourier Transform. Hence it is easier to use all of them for beamforming. Both the streams of 8 serialized signals are passed through weight multiplication. Beam data of weight multiplication

is added from both serialized streams to form a beam. Real and imaginary values are added accordingly. Performing addition in an FPGA design utilizes one more bit for full precision. This increases the number of bits in case of 64-channel data but serialized FFT output helps here. Vector accumulator has been used effectively, to sum up, multiple signals. This accumulation is of 8 serialized signals controlled by logic considering delays and casting of numbers.

Calculation of power of coherent sum is followed by truncation to 16 bits. There is a provision of scaling of beam value by gain inputs of power quantizer. Thus we get final coherent beam data to be recorded. This beam power is accumulated in a vector accumulator for instant beam plotting as of now. This is controlled by accumulation pulse of 1.3s duration. This is a quick check of beam band-shape observed run time. Proper implementation of bit file on FPGA and smoothness of band helps to know instantly. In this way, two beams have been included in the design. Both can be controlled independently via python scripts (refer fig.19). The final pipeline of beam data would be developed later with beam data on 10GbE port.

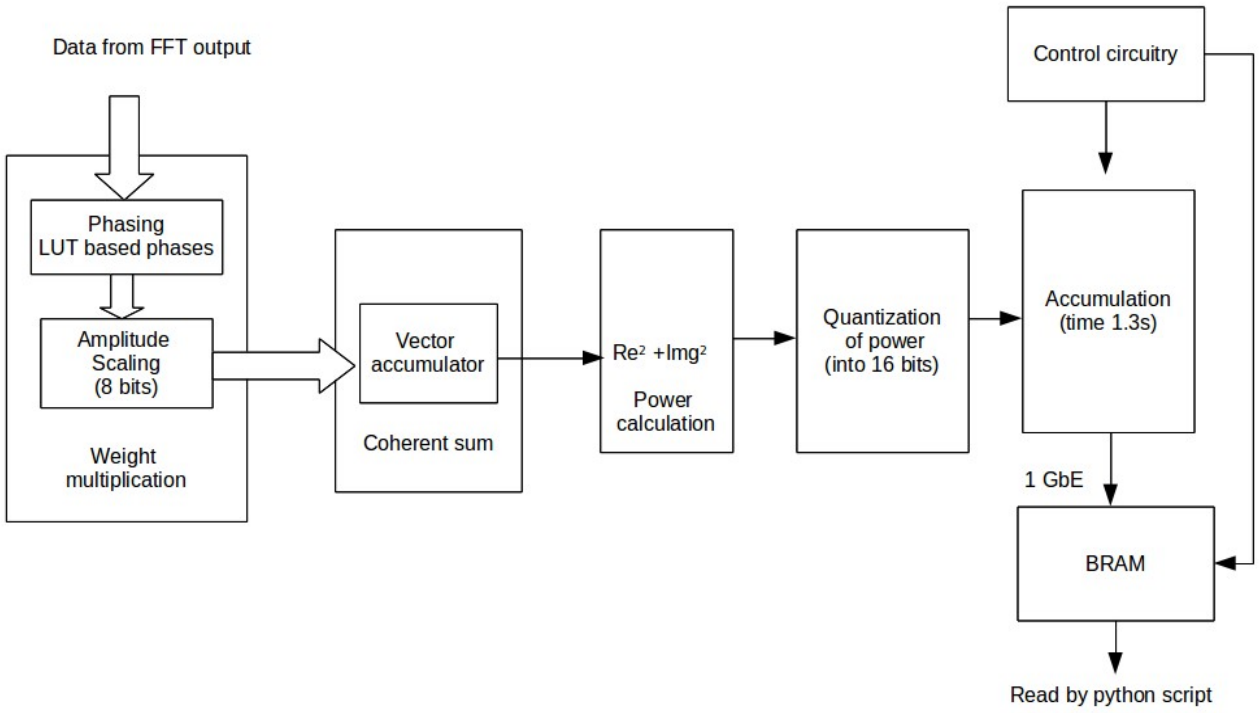


Fig.19 Beamformer chain

Concluding the current status, 16 input, single FPGA board, 2 coherent beam, 15 pairs of cross-correlations, 16 auto-correlations design is ready as first basic version and it is being tested for various cases and modifications.

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

During this development, 8-element, 12-element design were prepared as an initial trial. This intermediate experiments are also mentioned appendix-C. The design logic explained in earlier chapters was used in those experiment. 8 inputs of serialized FFT streams were used for correlation by de-serialization. 7 pairs of correlation were used and phase correction was done creating 2 coherent beams. Results of that design are also included in appendix-C. These intermediate designs very helplful to achieve 16-input design.

16-element narrowband beamformer:

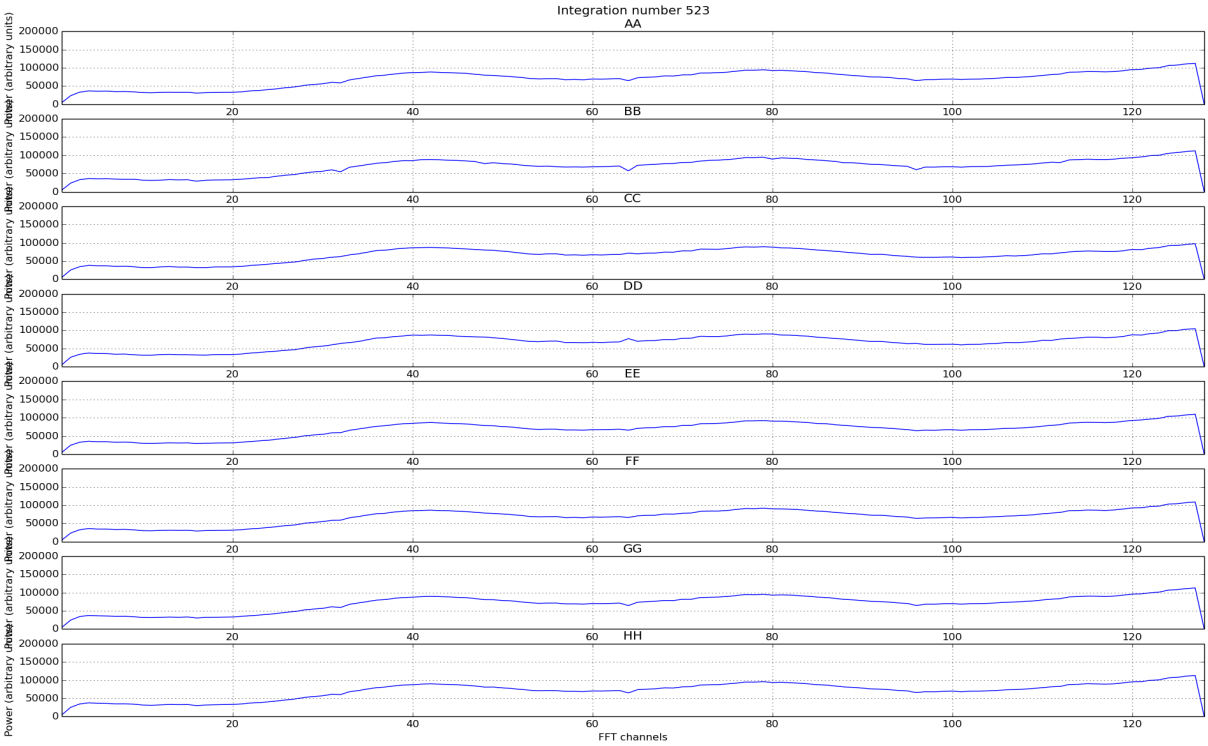
FFT block for 64-channel ADC data accepts minimum 16 inputs for Fourier transform. After solving erratic phase of correlation [refer appendix-C, 12-element design] in some of the pairs of inputs, 16 input design has been developed. Primarily it has been tested with identical signals obtained through power divider. Broadband noise source followed by 25 MHz LPF has been used. This design is being tested for various cases of correlation and beamforming. There is an independent control for two beams. Amplitude/phase can also be controlled independently. As of now, 8 unsigned number is used for scaling. There is a truncation of bits after phase multiplication and amplitude scaling. A 36-bit complex number is truncated to 8 bits for correlation and calculated power of beam data is truncated to Fix_6_5 bit format. There 15 pairs of cross-correlation and 16 auto-correlation at a time. There is design block which provides facility to correlate any combination of two elements out of 16, when the design is running. This provides quick & specific debugging of correlation. Both the beams are having different signal streams, so there is a selection between which cross-correlation is to be observed. Correlation is stored in BRAM and read with the integration of 0.6s whereas beam data is updated every 1.3s into BRAM. Design blocks and results are shown in following figures.

 # Design:

This design is conveniently developed so that scaling to a higher number of inputs would be easier. Designs modules are grouped in an optimum way to expand volume or debug the design issues. FFT points used in this design are 256, so results plots show 128 FFT channels. This experiment has been scaled in terms of FFT points and the number of beams. As a primary test, it works in case of correlation of a pair of antenna elements and phase correction. Fig.20 shows auto-correlation of 16 inputs divided into 8 each. Fig.21 shows cross-correlation correlation phase before and after phasing.

 # Auto-correlation of 16 input design:

(a) For first 8 inputs (A-H):



(b) For next 8 inputs (I-L):

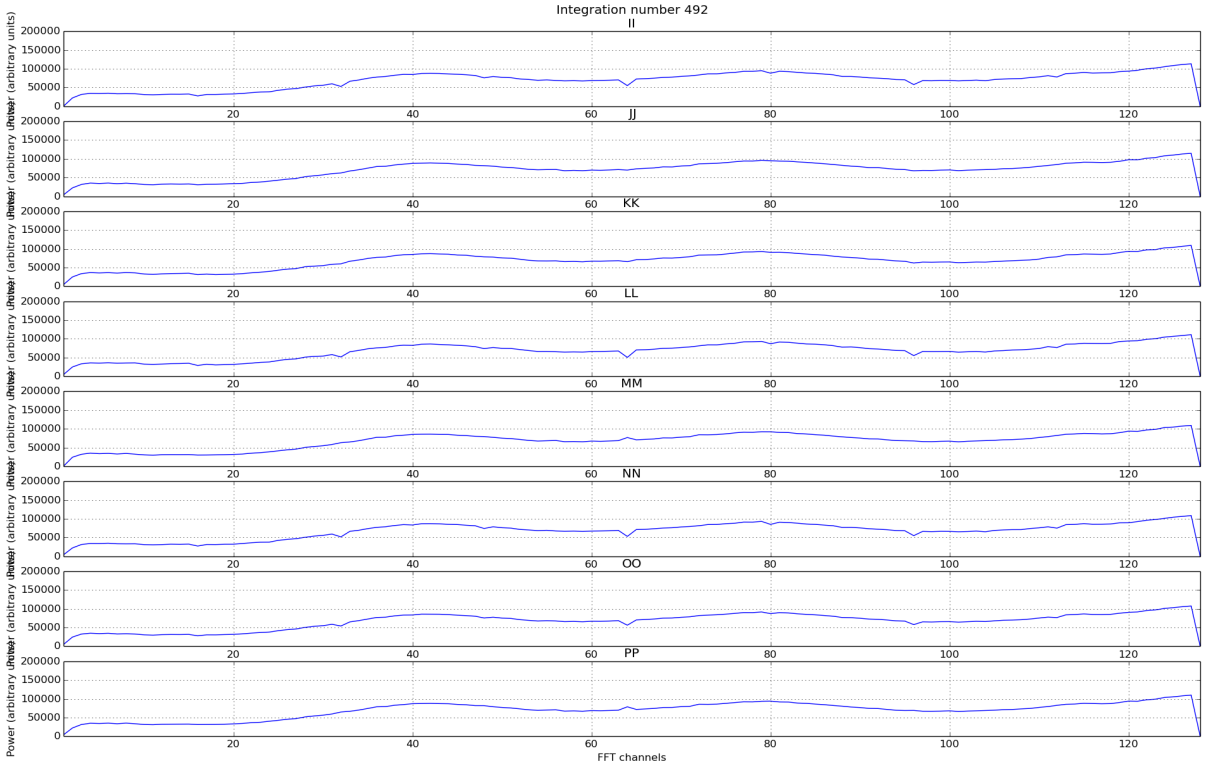
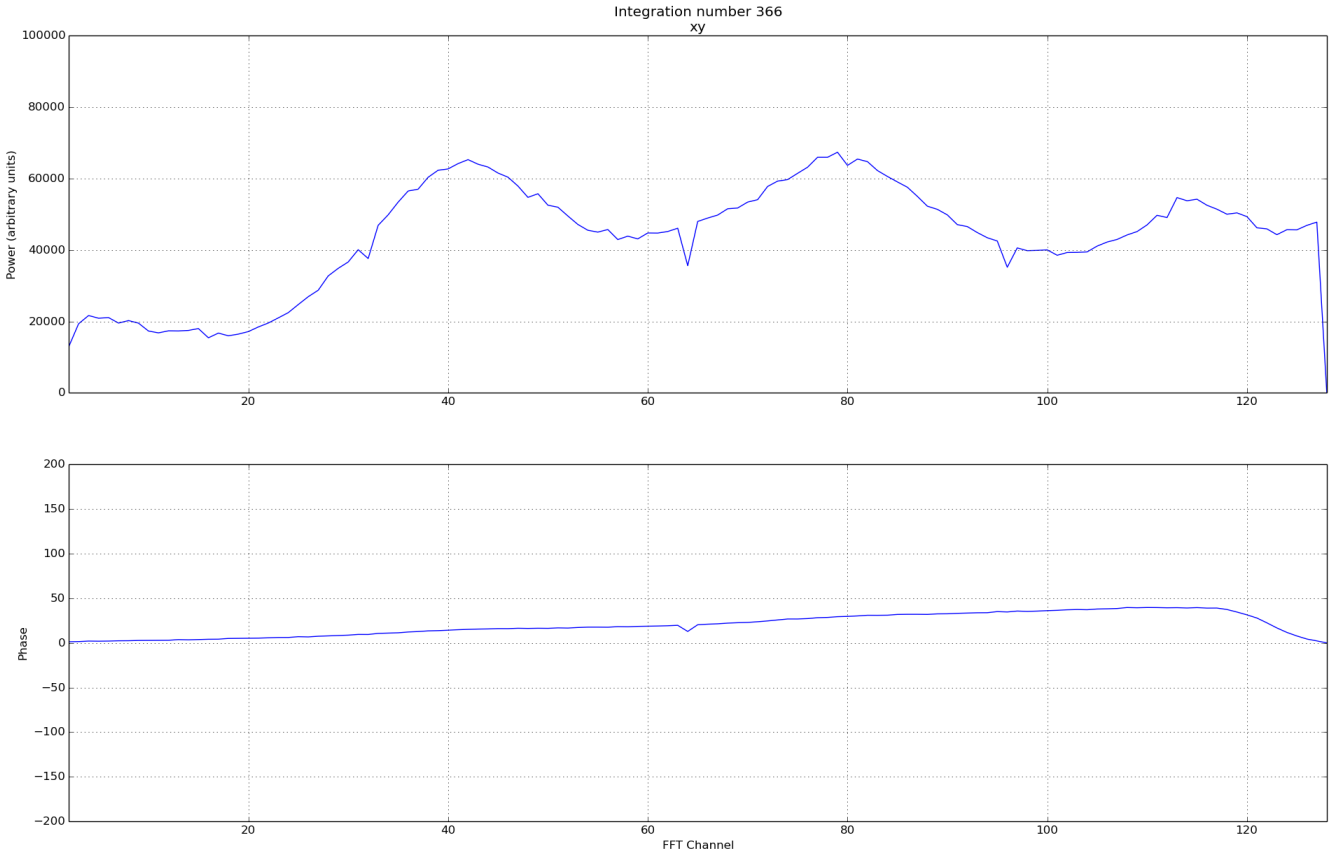


Fig.20 Auto-correlation of 16 inputs, 128 FFT channels on the x-axis, power on the y-axis in arbitrary units

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

 # Correlation of a pair after phase multiplication:

(a) For this pair of correlation, there is a mismatch of cable, so there is phase initially:



(b) After Phasing:

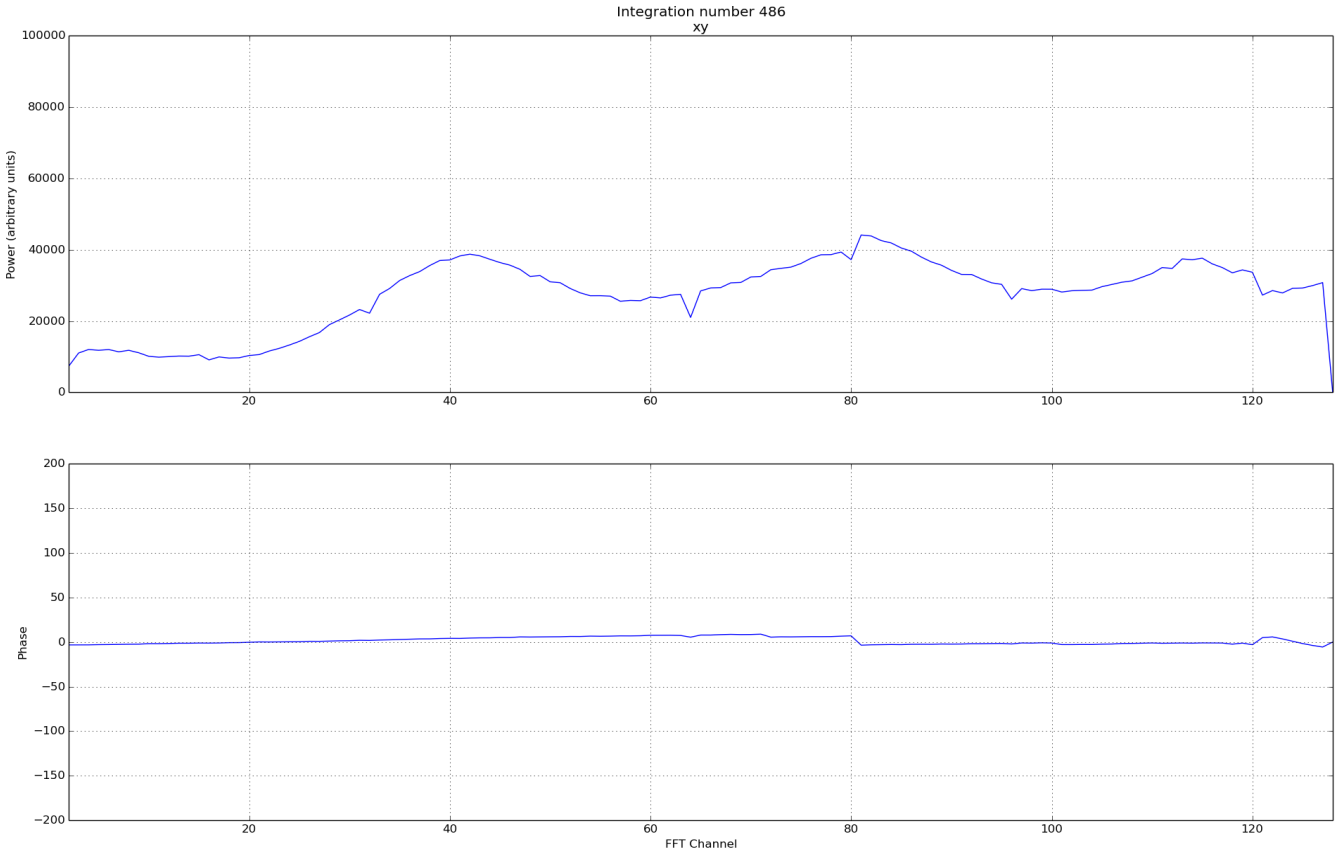


Fig.21 Phase correction of a pair with a different cable length of cross-correlation

Scaling of a basic version of 16-input beamformer:

The design discussed earlier is the reference design for 64-channel ADC based, 16 input, FPA beamformer. This single board design uses FFT size as 256, calculates auto-correlation of 16 signals, 15 cross-correlation pairs and forms 2 beams observed by python scripts. So, the results and changes are quickly observed when a script runs. As this is a proof-of-concept for actual designs, some experiments were carried to scale this basic model. In FPGA based designs, there are limitations of resources utilization and timing accuracy. This is quite clear from comparisons shown in fig.22. There are 3 dimensions of scaling the basic version viz. increment in the number of inputs, number of beams, number of cross-correlation pairs. Staring from reference, FFT size has been compiled from 256 to 2048 points successfully. The number of beams has been increased to four with independent control of phasing and amplitude scaling. These two dimensions have been compiled and tested with sample signals as well.

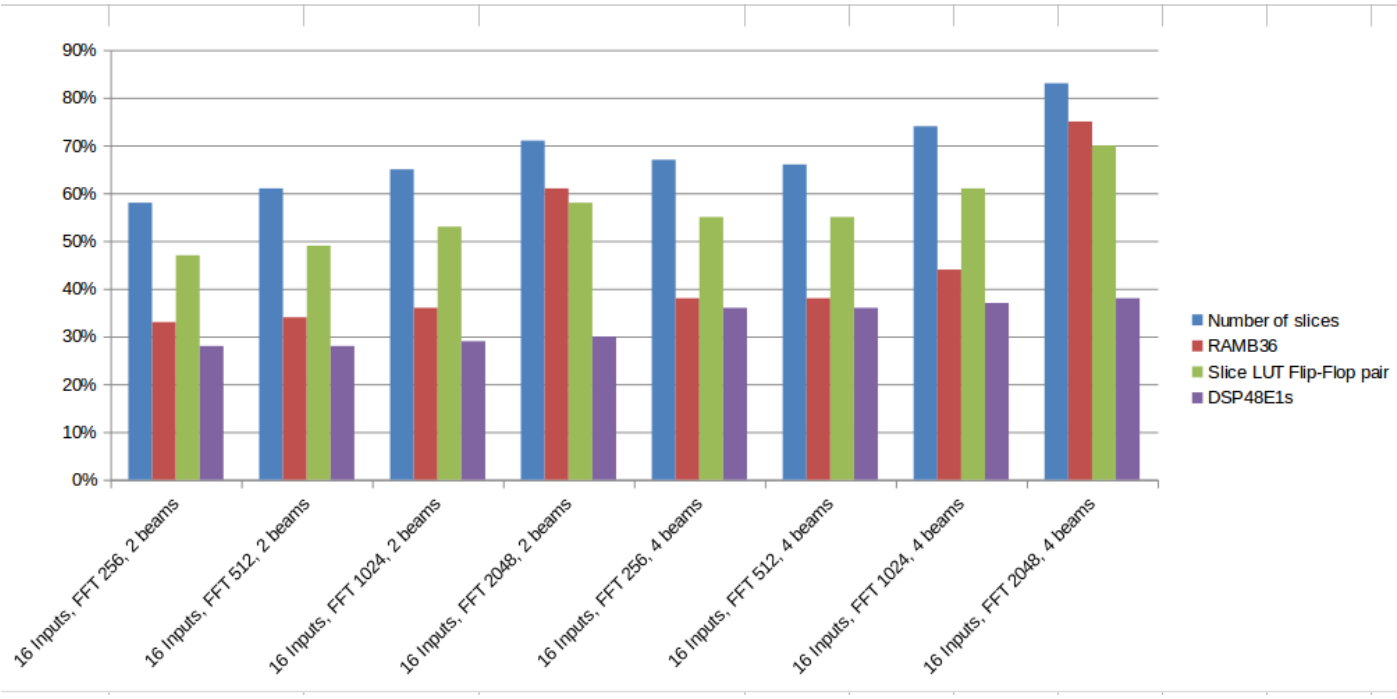


Fig.22 Resource utilization of scaled designs

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

The Problem of dips in beam band-shape:

The basic version of beamformer was scaled to 2048-point FFT and 4 beams. Testing this version showed dips in beam band-shape while testing with noise source. Fig.23 shows dips at regular intervals located at submultiples of N, N/2, N/4... where N=point FFT is computed. This problem was solved by bit shifting before FFT input. FFT uses 18-bit coefficients whereas digitized inputs are 12-bit samples just. These 12 bits were cast to 18 bits and shifted to left (MSB), This solved the problem of dips, as well as a weaker noise signal, could be observed easily compared to earlier testing.

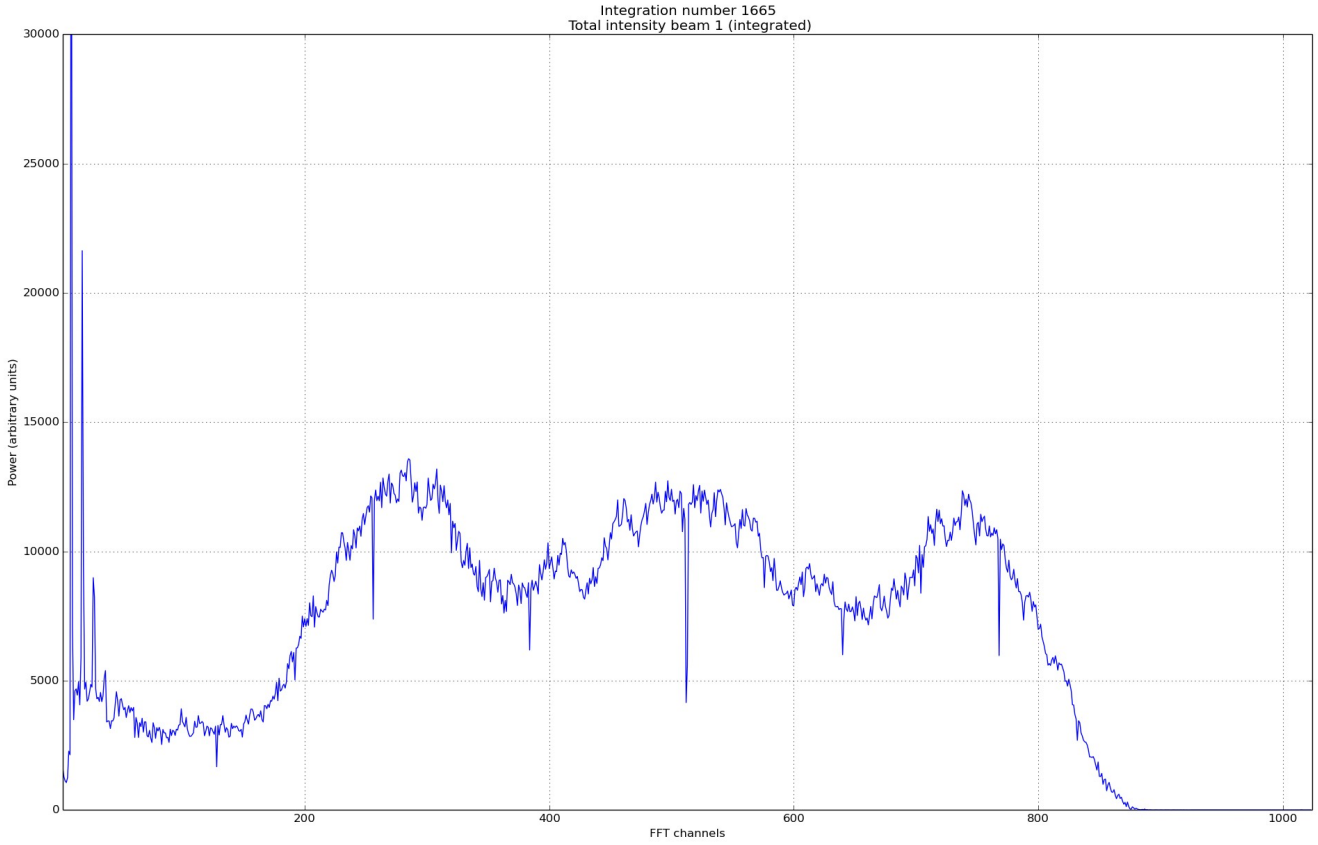


Fig.23 Noise beam band-shape; shows dips at regular channels

The problem of ramps cross-correlation phase:

Cross-correlation phase of two signals showed multiples ramps as shown in fig.24. This happened because some samples were missing while scaling FFT points. This was solved by increasing buffer size used in reordering according to FFT length.

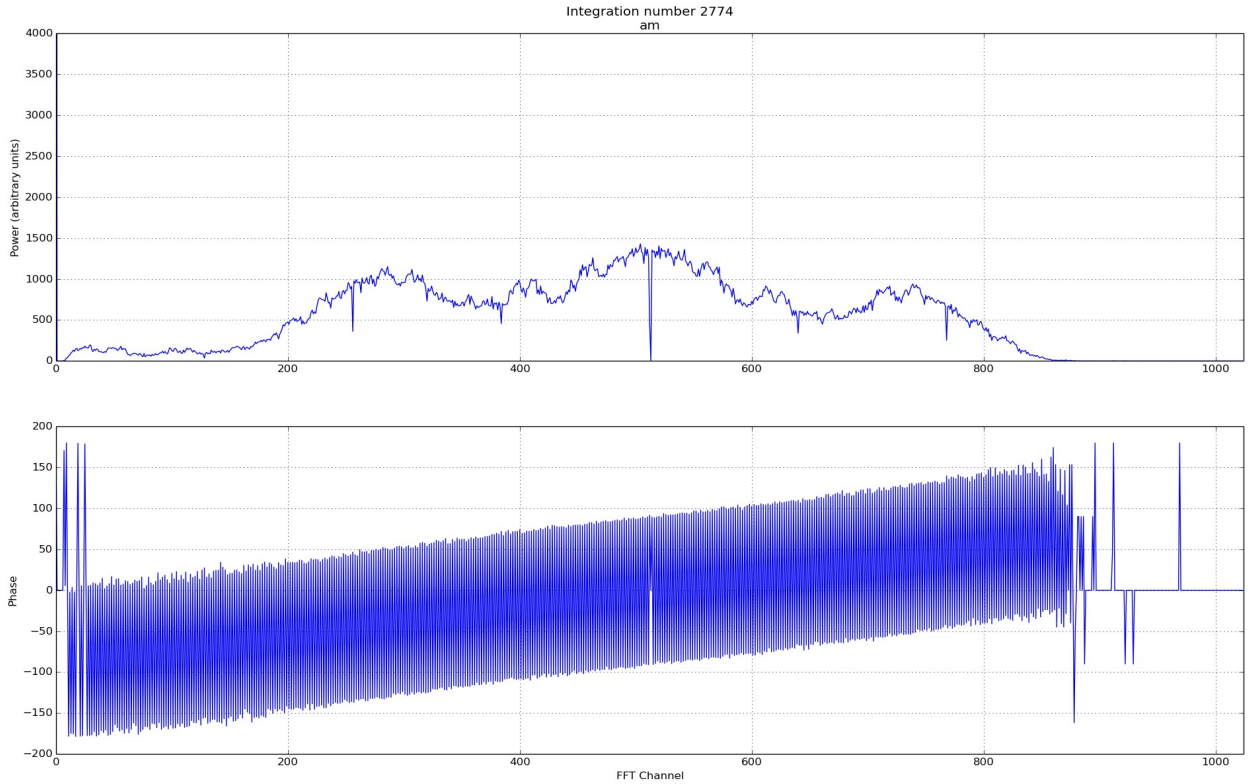


Fig.24 The fixed delay introduced due to samples ignored in reordered data

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

Chapter 7. 16-input narrowband FPA beamformer design

As mentioned in the chapter 6, the basic design has been scaled in terms of FFT size and number of beams. Now, we have x64_adc based, 16 input 4 beams design with FFT size 2048 points. This latest design is being tested for further experiments. In basic design, an 8-bit unsigned number is used for amplitude scaling and power calculated in beamforming was rounded to Fix_6_5 bits before integration for python scripts. Amplitude scaling is done to ensure tapering can be provided among FPA elements. When a complex value is multiplied by the 8-bit number(256 value), tapering can be provided up to 24 dB. In basic design, scaling was not observed exactly to the expectation due to the assumed way of truncation. Improper scaling and saturation at some point were observed. On performing some experiments on methods of truncation and rounding, an expected design has been developed now. This latest design does exact scaling of signal up to 255 values with little bit saturation for higher power and gains used in the design. Earlier only 6 bits were used to round the beam power, now 16 bits are used instead. This working has been tested with CW and noise signal.

#Features of latest version:

- 16 inputs, 4 beams, FFT size: 2048, 15 Cross-correlation pairs
- FFT output: (18bit Re+ 18bit Img)
- Quantization used for correlator: (4 bit Re + 4 bit Img)
- Beam power rounded to 16 bits before integration done for python scripts.
- Phase resolution : 5.68 value per degree (360 degrees mapped to 2048)
- Amplitude scaling : 8 unsigned bits used, up to 24 dB tapering possible
- Integration time for correlation : 0.67s
- Integration time for beam : 1.34s
- Frequency resolution: 31.25 KHz for 32MHz bandwidth & 1024 FFT channels
- Generic python scripts which use parameters: FFT size, beam number, scaling factor, steering angle, correlation pair etc.
- Simultaneous observation of all beams possible
- Flagging of bad signals is possible by zero scaling factor in the tapering script.
- The Gain of a beam can be controlled depending upon input power level.
- This latest design does exact scaling of signal up to 255 values with some saturation for higher power and gain used in the design.

#Experiment set up:

- 64 channel ADC interfaced to ROACH-1 board which performs both correlation and beam-forming both in the same design.
- CW input used: 15MHz, power -46dBm (signal generator's RF output given through attenuators & power dividers via 2-3m coaxial cables)
- Noise inputs used: broadband noise source + 25 MHz LPF, power: -54dBm (applied through power dividers via 2-3m coaxial cables)
- Mapping of external connection box to design inputs is explained in appendix-E.

#Design block diagram:

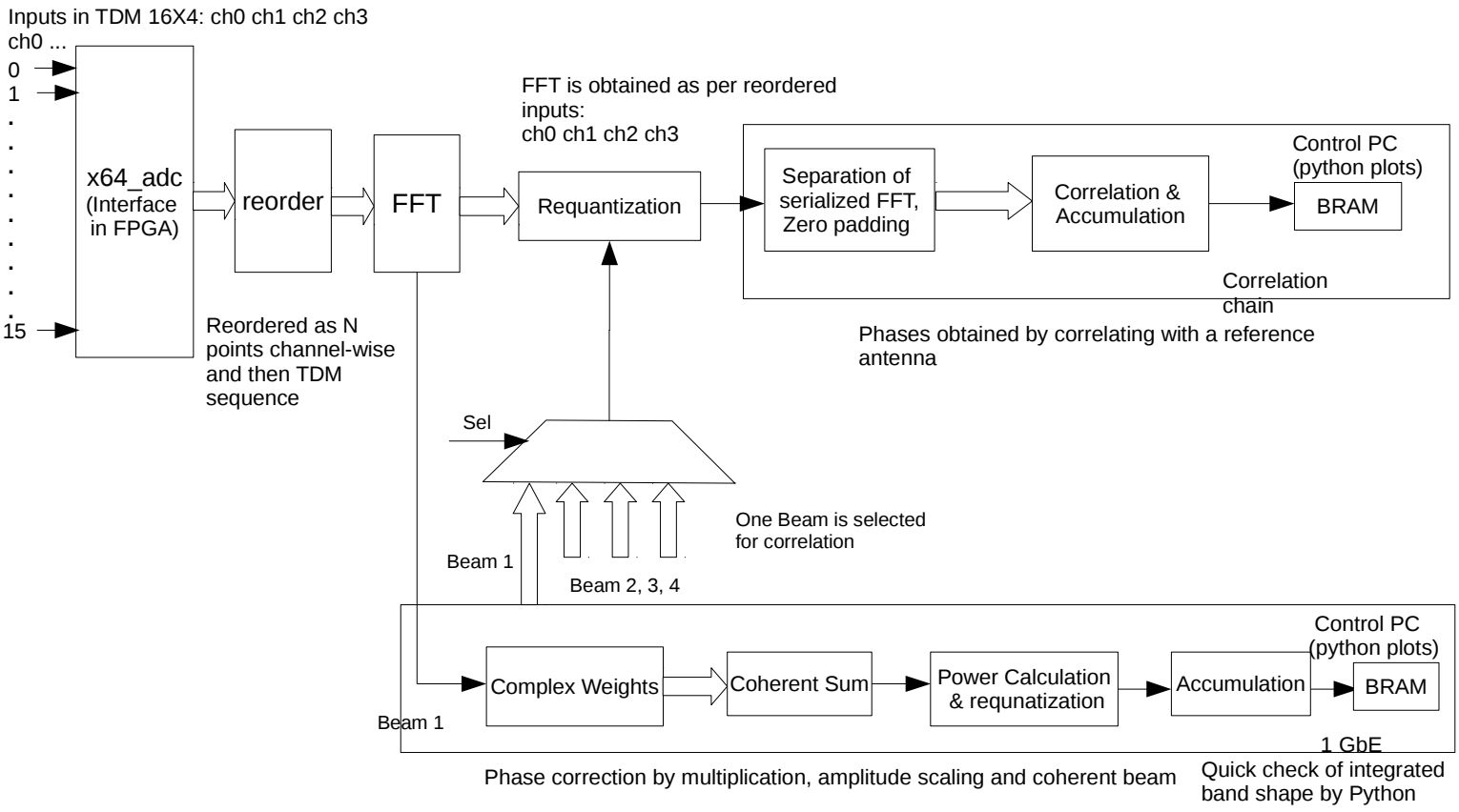


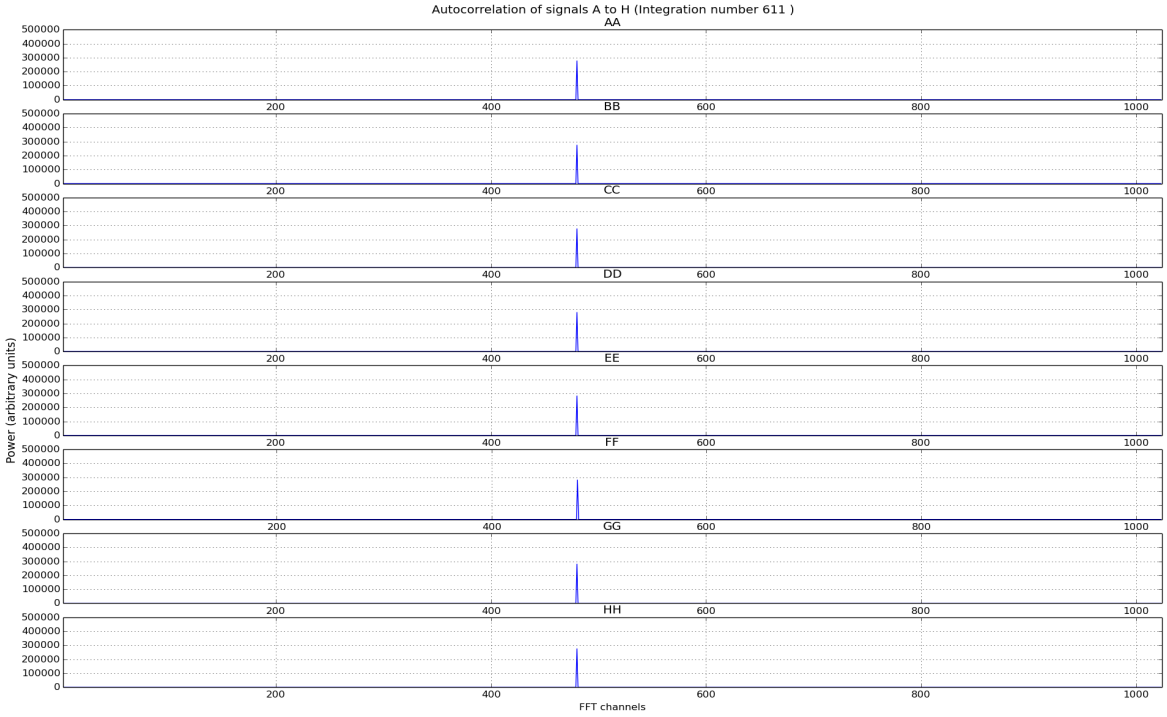
Fig.25 FPGA design block diagram of Narrowband beam-former

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

i. Testing with CW inputs

#Auto-correlation of 16 input design:

(a) For first 8 inputs (A-H):



(b) For next 8 inputs (I-L):

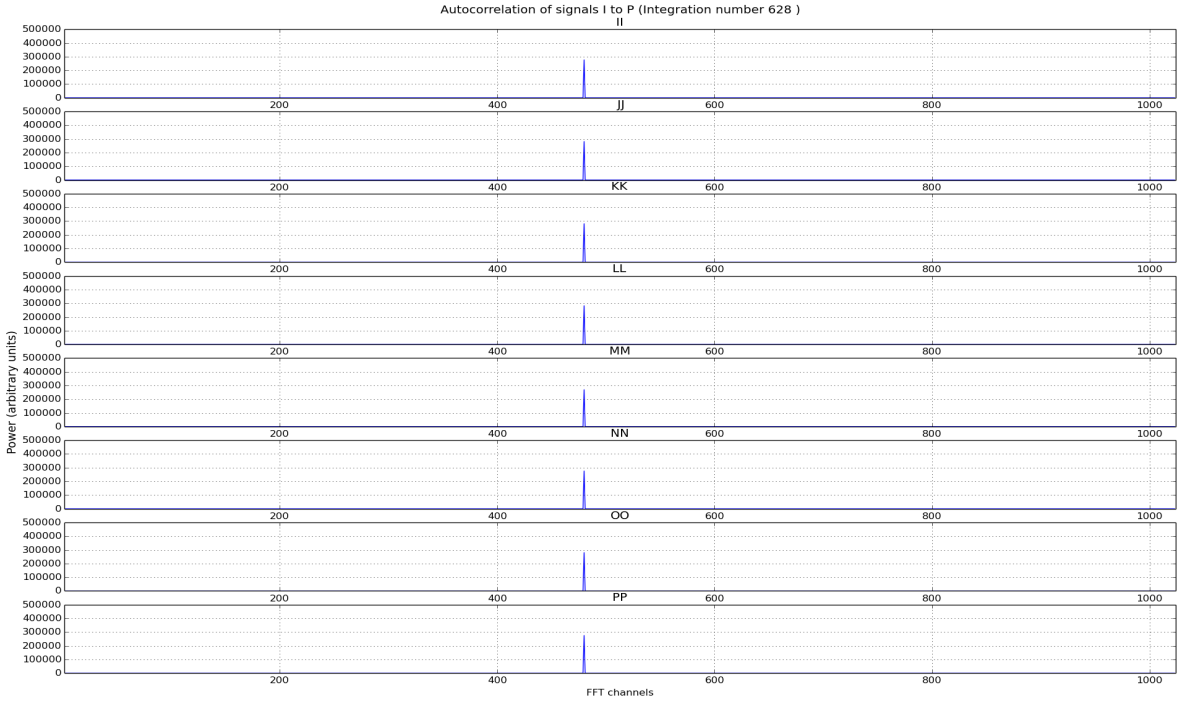


Fig.26 Auto-correlation of 16 inputs, 1024 FFT channels on the x-axis, power on the y-axis in arbitrary units

#Cross-correlation of a pair of 16 input design:

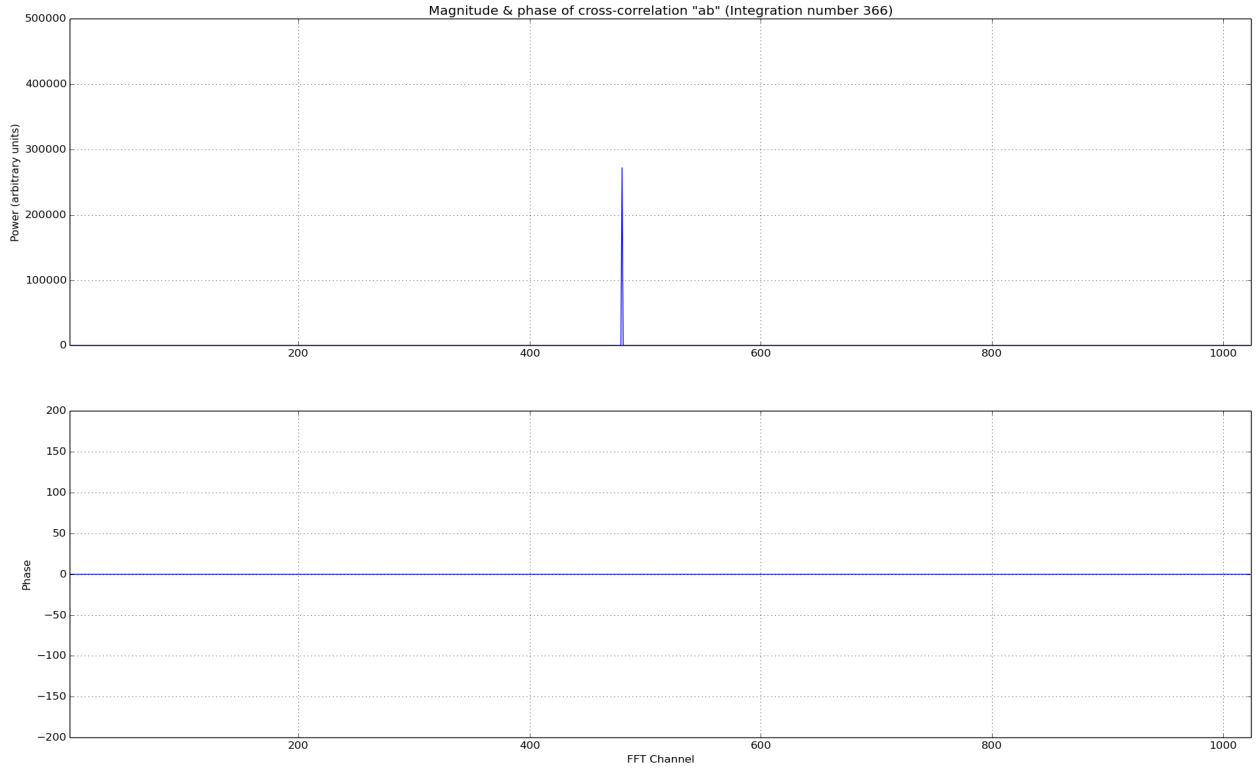


Fig.27 Cross-correlation of a pair where cable length is same

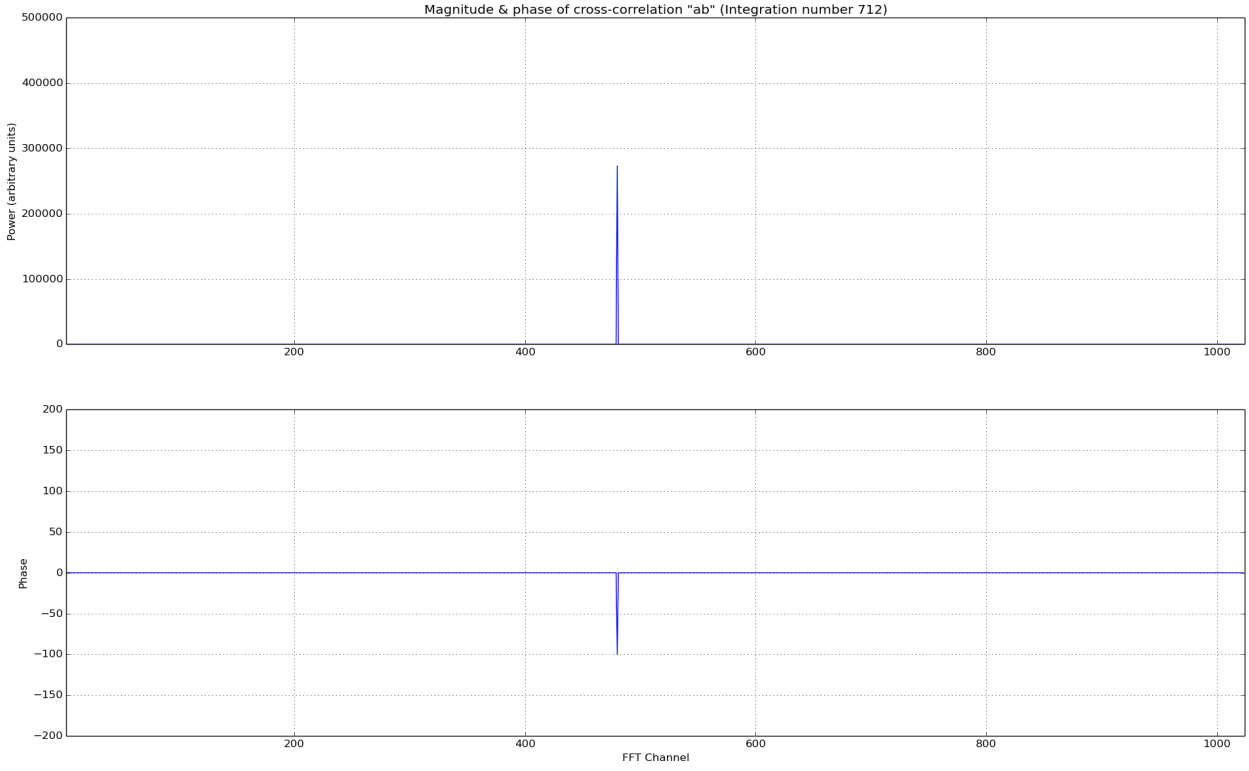
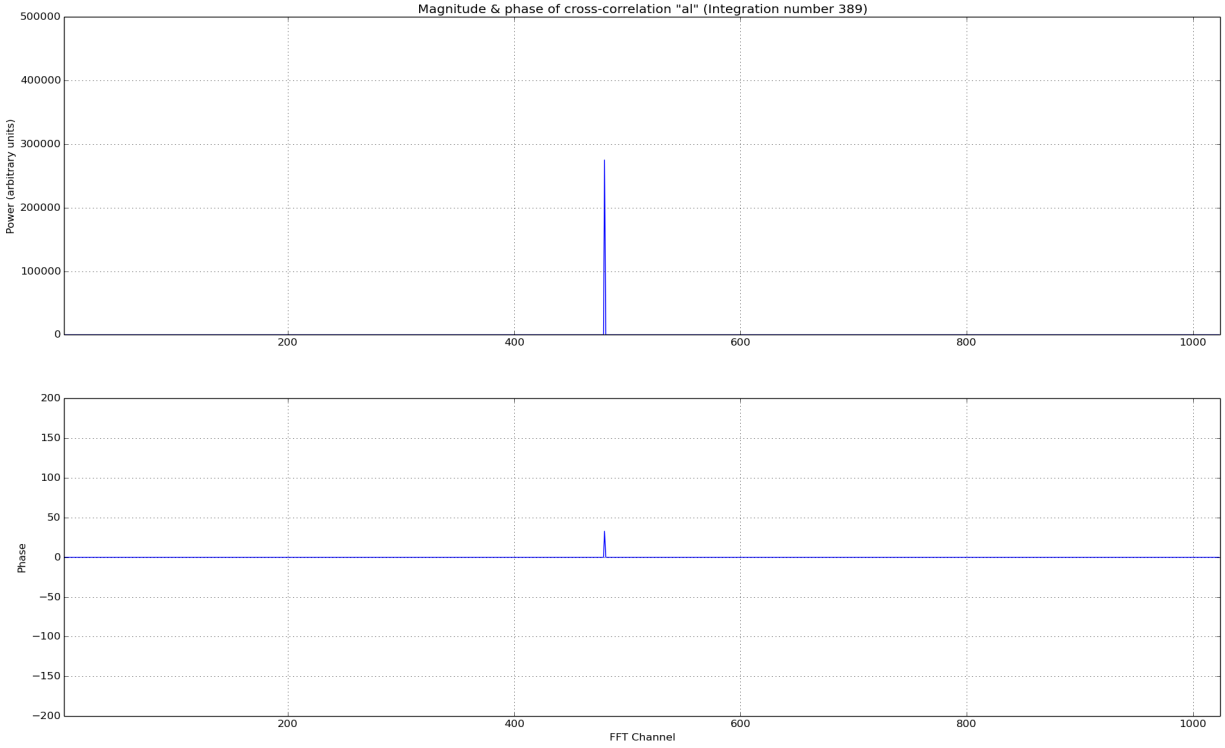


Fig.28 Cross-correlation when an explicit angle (-100) is provided to an input by script

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

#Correlation of a pair after phase multiplication:

(a) For this pair of correlation, there is a mismatch of cable, so there is phase initially:



(b) After Phasing:

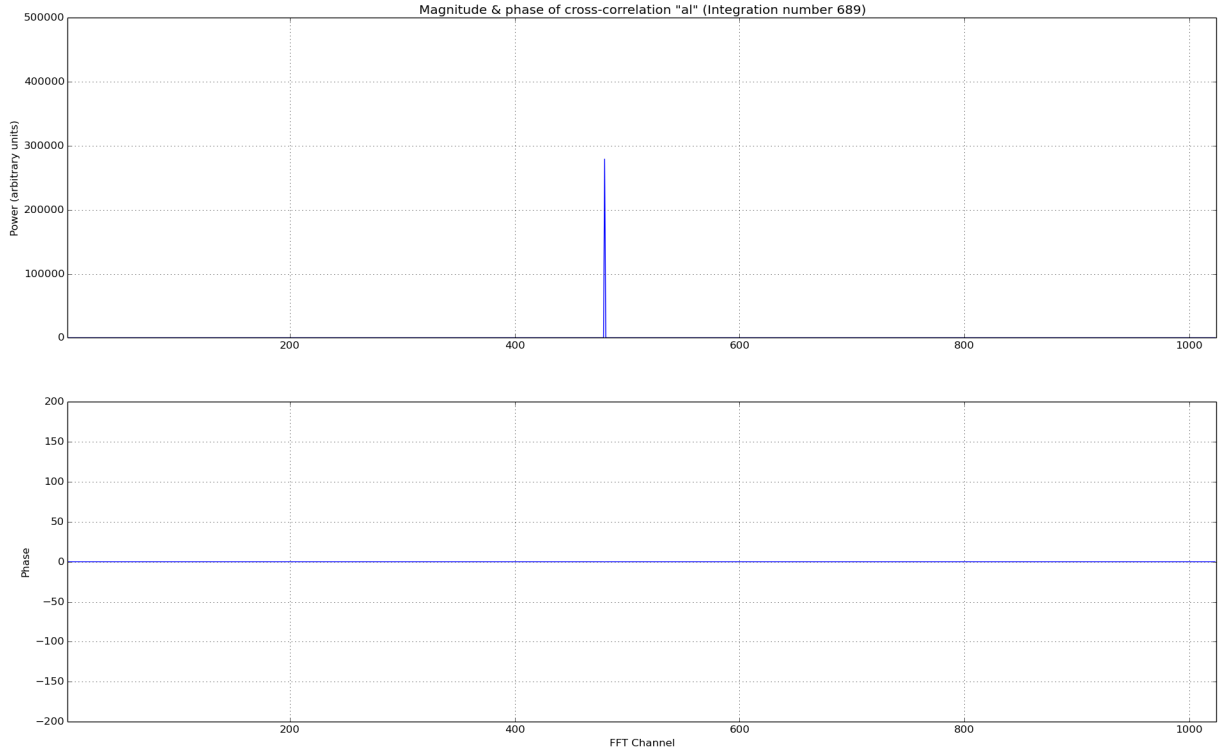
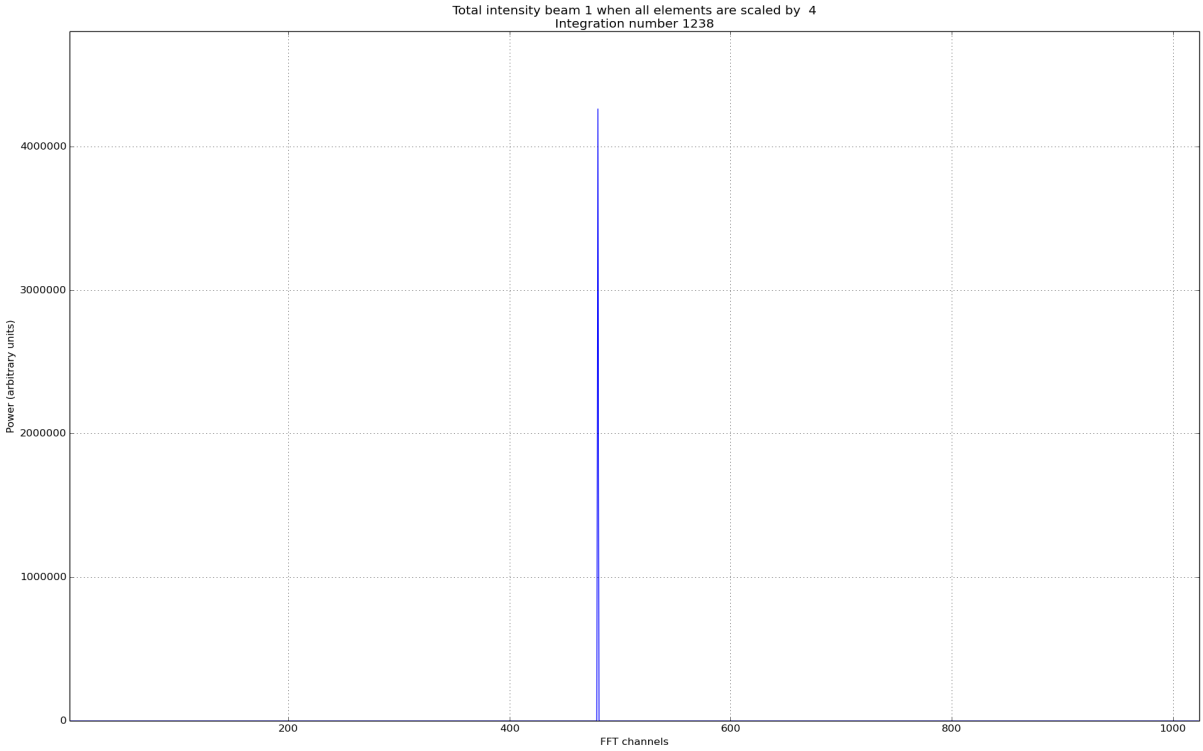


Fig.29 Phase correction of a pair with a different cable length of cross-correlation

#Beam data of 16 input design:

(a) beam1 :



(b) Beam 2:

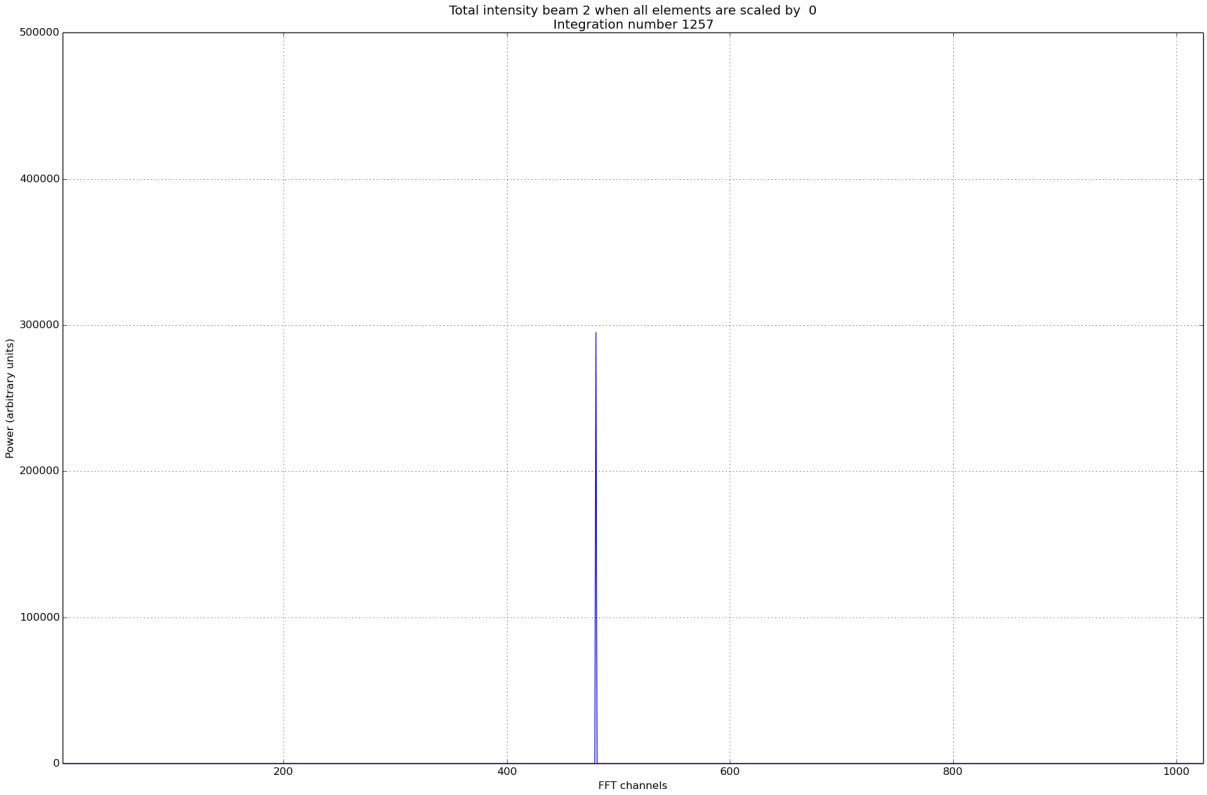
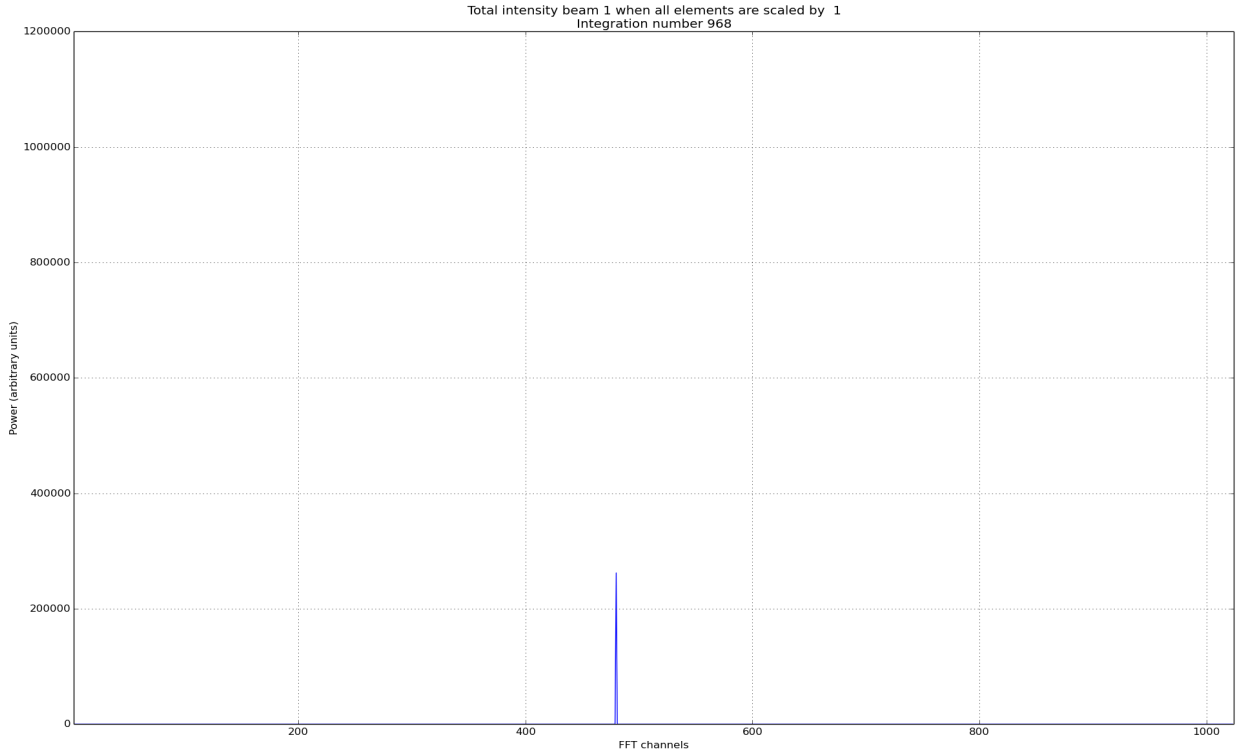


Fig.30 Different beams can be observed in parallel

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

#Testing of amplitude scaling for tapering:

(a) Amplitude scaling factor: 1



(b) Amplitude scaling factor: 2

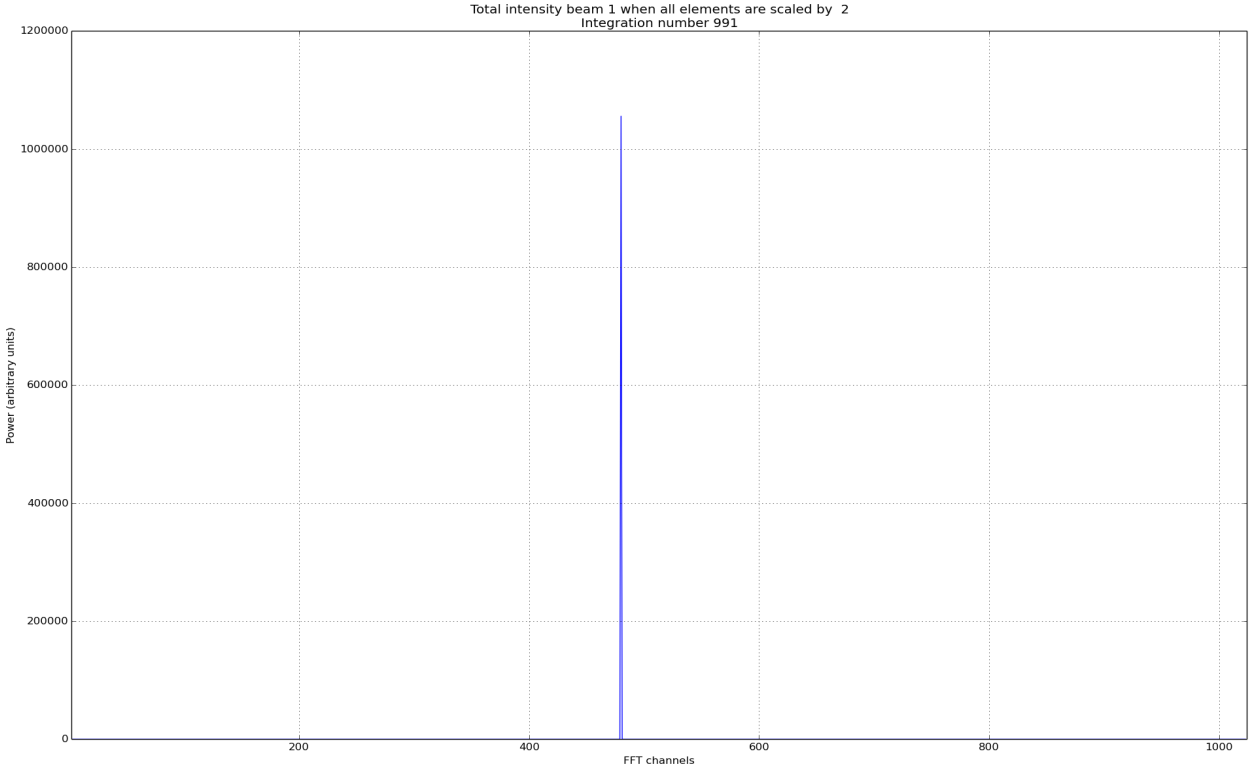


Fig.31 Effect of amplitude scaling 1 to 2; Y-axis range changed based on the value

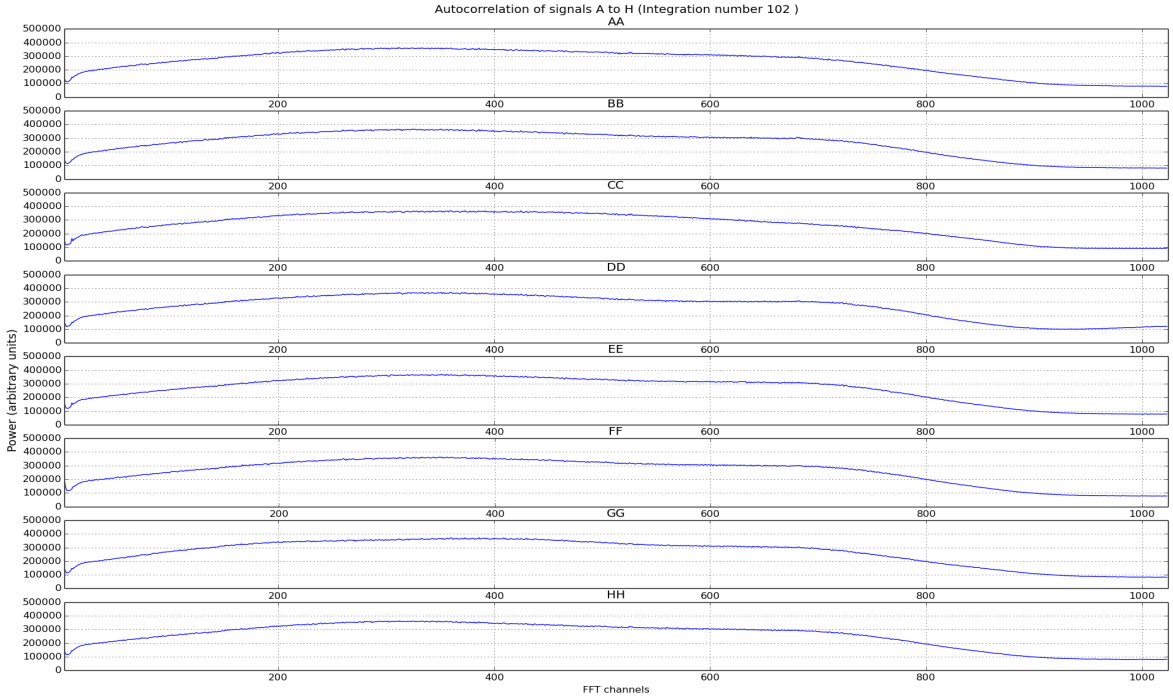
Observations of testing with CW input:

- Carrier wave of 15 MHz is observed at 480 th FFT channel as expected for 32MHz bandwidth and 2048 point FFT. (fig.26 to fig.31)
- There is no cross-correlation phase for a pair (AB) of signals when cables used have same lengths (fig.27)
- With identical signals, there is no phase difference in inputs. Phase can be applied to one of the input by script e. g. -100 degree was applied in one of the signal (fig.28)
- For "AL" element pair of cross-correlation, there is phase due to difference in cable length, which gets corrected on phase multiplication (fig.29)
- Total intensity beam has been integrated for 1.3s and plotted by python scripts. Multiple beam band-shapes can be observed simultaneously. (fig.30 (a) & (b)) plotted in different terminals)
- Correlation chain is common for all beams. Data going into correlation chain can controlled. Phasing script takes care of appropriate beam data based on beam_number parameter passed.
- Scaling inputs by 2 reflects increased beam power by 4 times as observed on Y-axis in fig.31(a) & (b)

ii. Testing with noise inputs

#Auto-correlation of 16 input design:

(a) For first 8 inputs (A-H):



(b) For next 8 inputs (I-L):

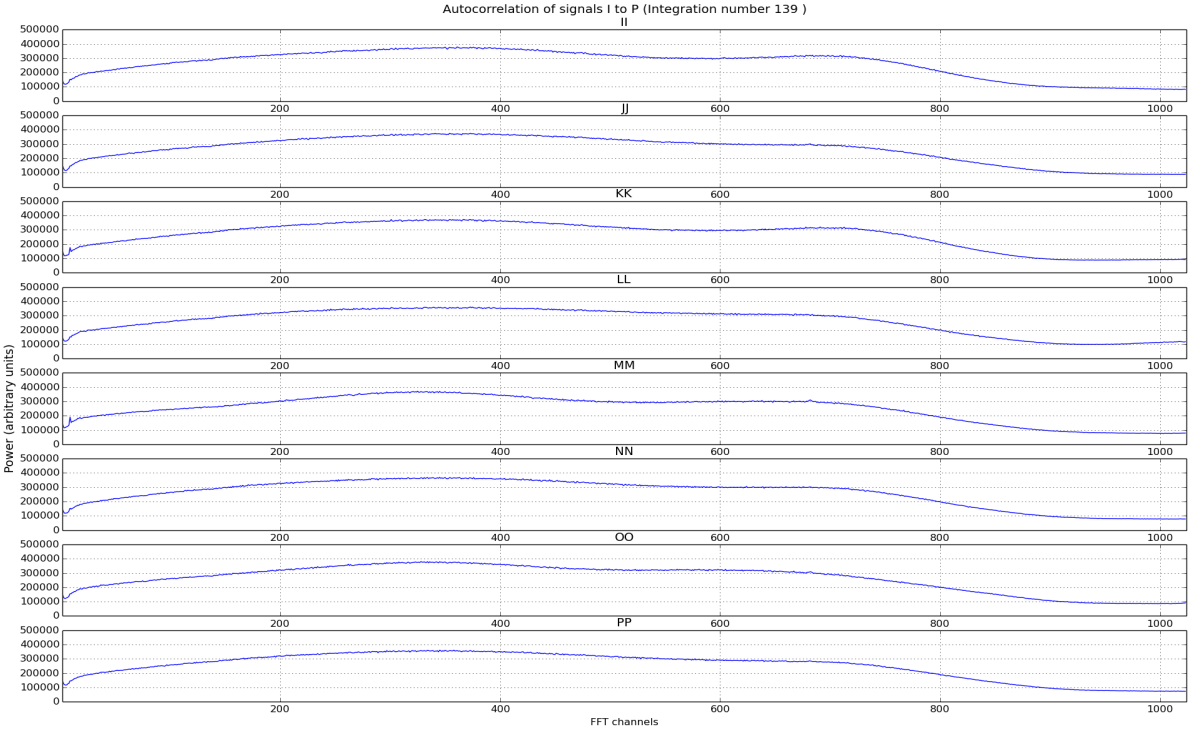
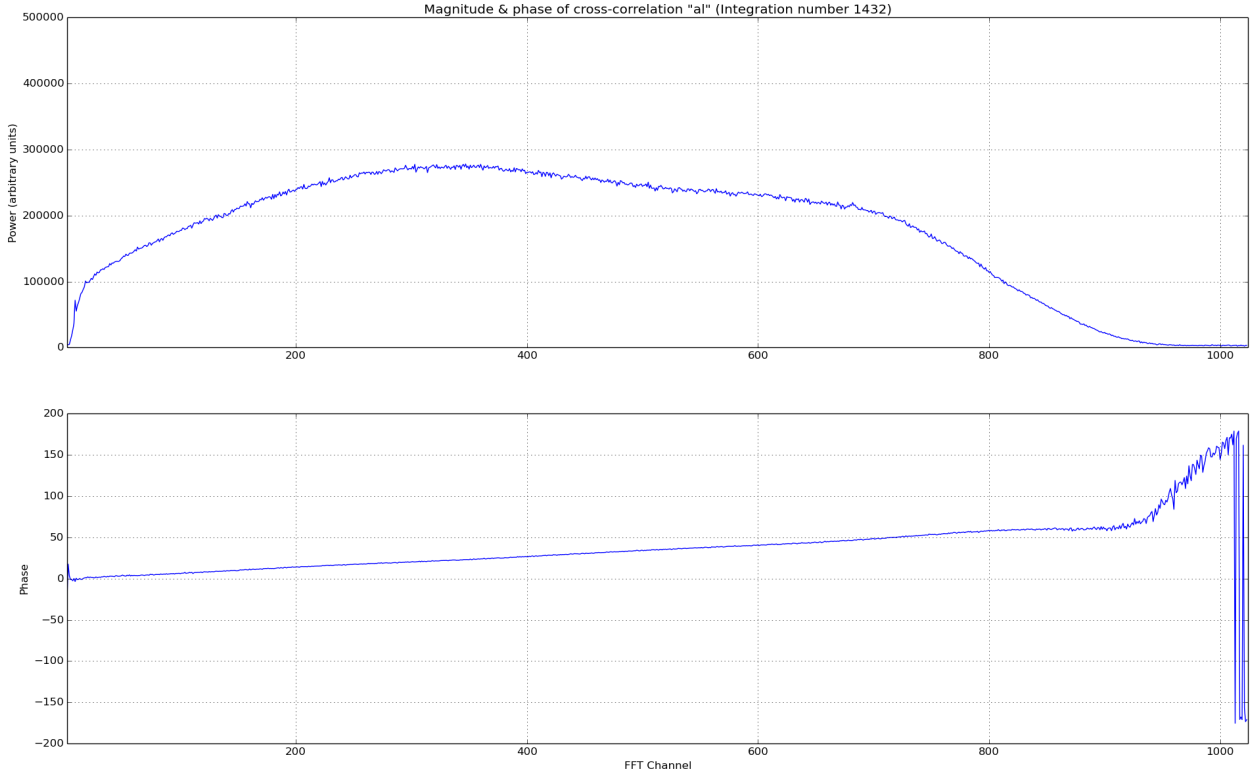


Fig.32 Auto-correlation of 16 inputs, 1024 FFT channels on the x-axis, power on the y-axis in arbitrary units

 # Correlation of a pair after phase multiplication:

(a) For this pair of correlation, there is a mismatch of cable, so there is phase initially:



(b) After Phasing:

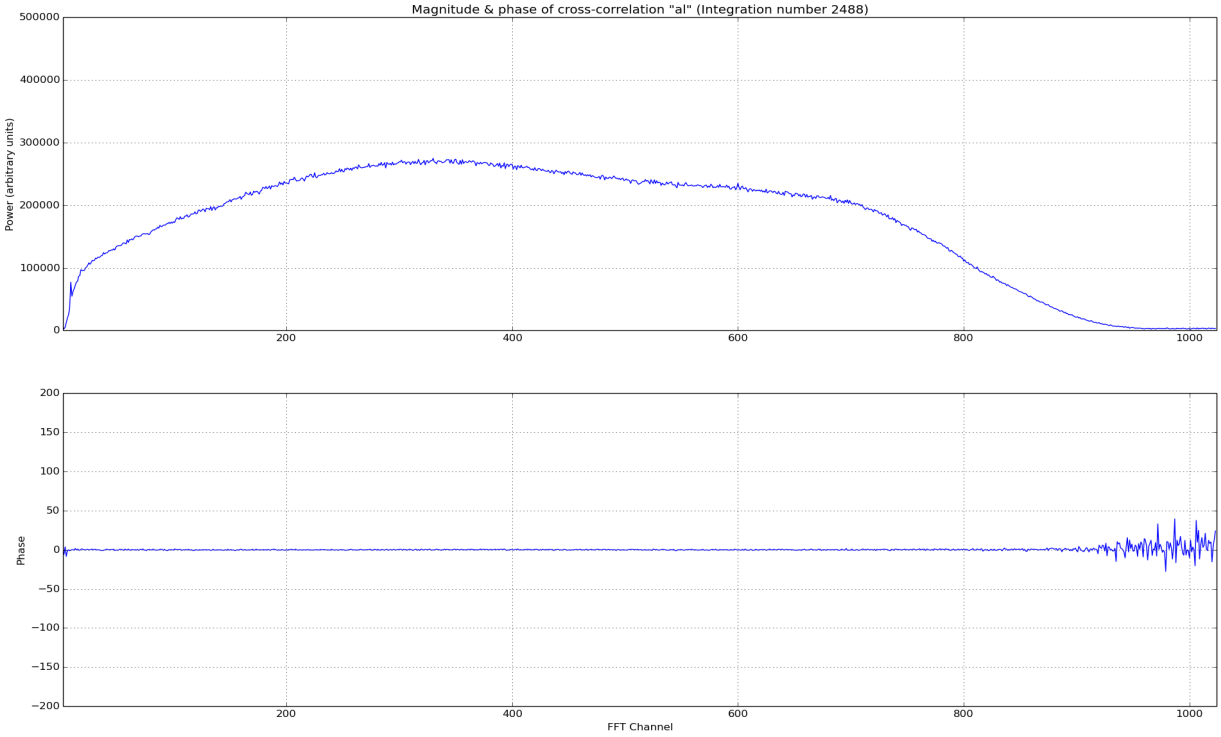


Fig.33 Phase correction of a pair with a different cable lengths

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

 # Cross-correlation of a pair of 16 input design:

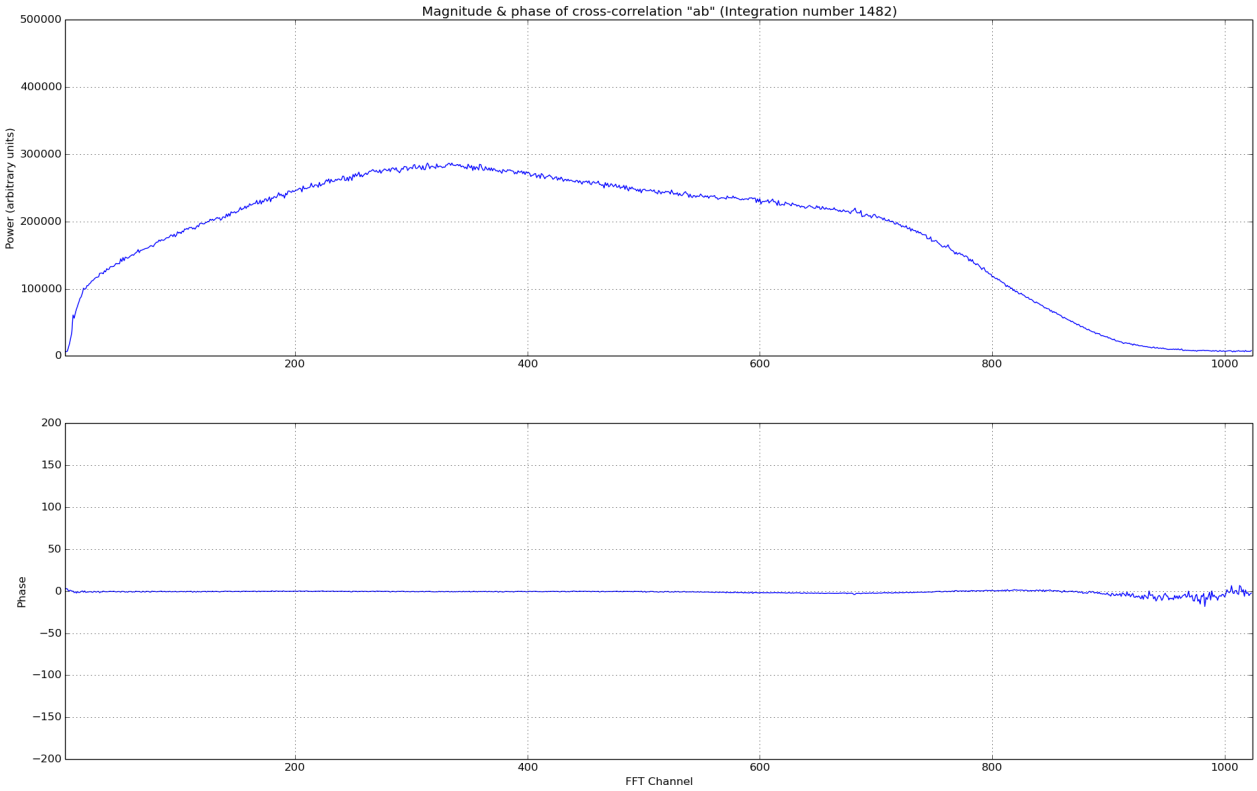


Fig.34 Cross-correlation of a pair where cable length is same

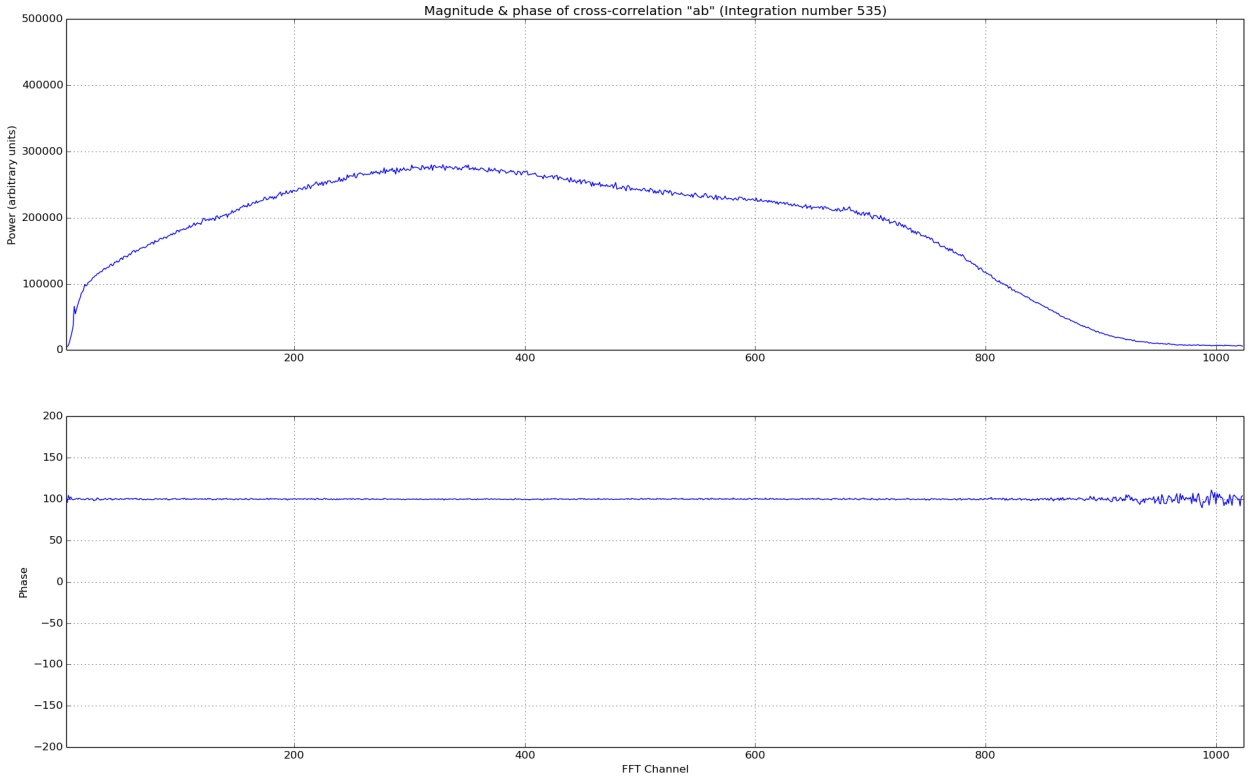


Fig.35 Cross-correlation when an explicit angle (+100) is provided to an input by the script

 # Beam data of 16 input design:

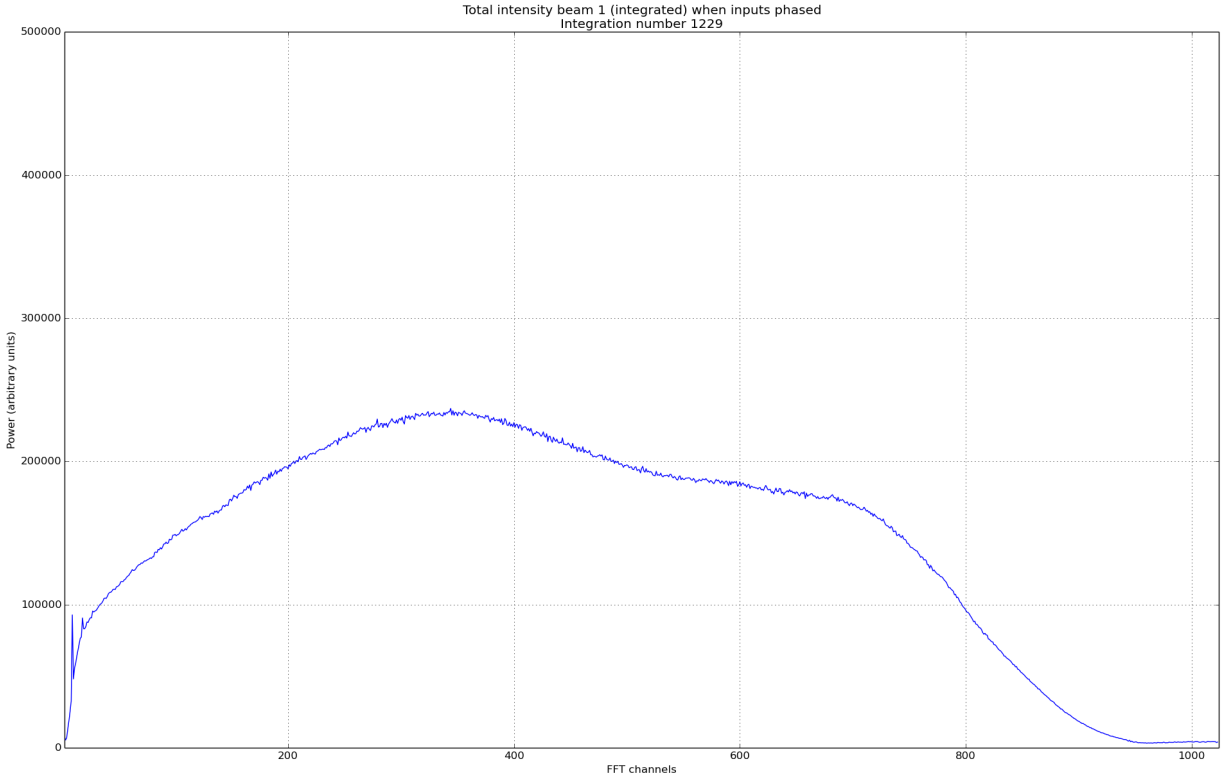


Fig.36 Beam band-shape of beam 1

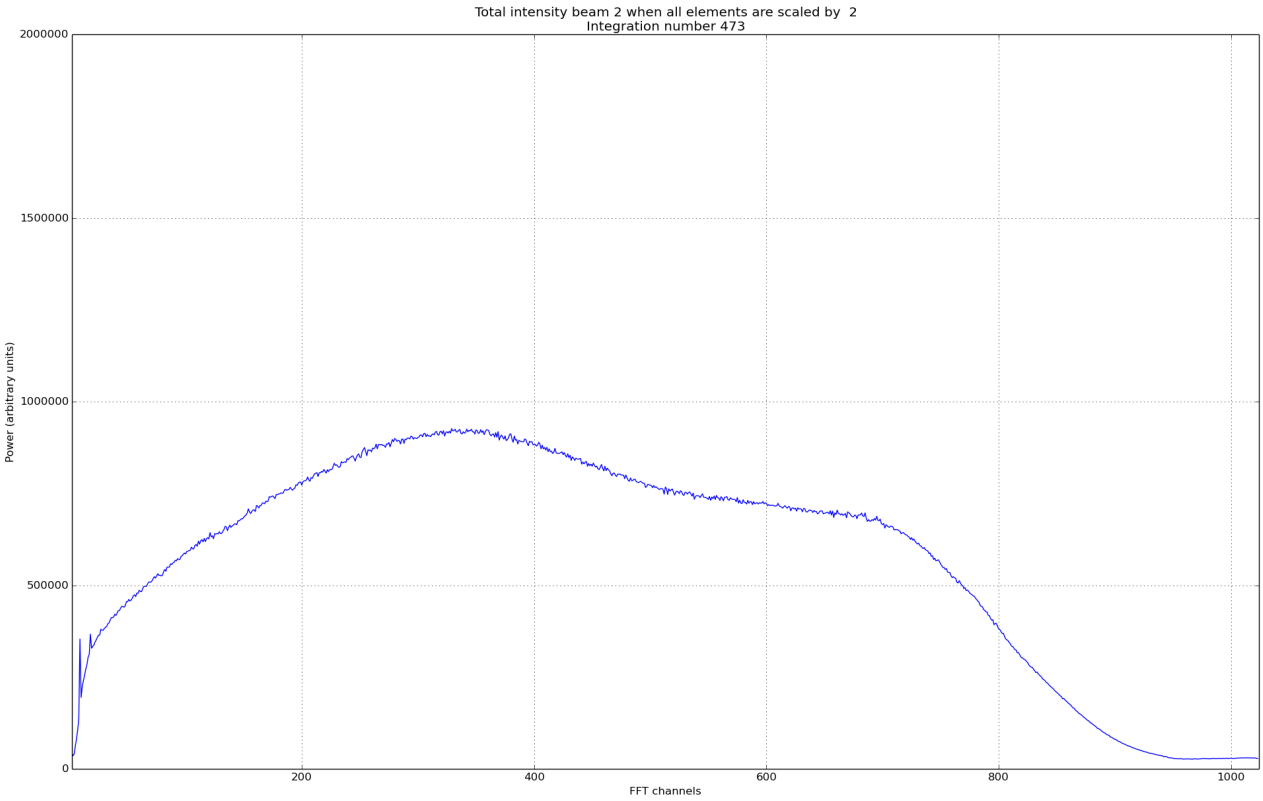
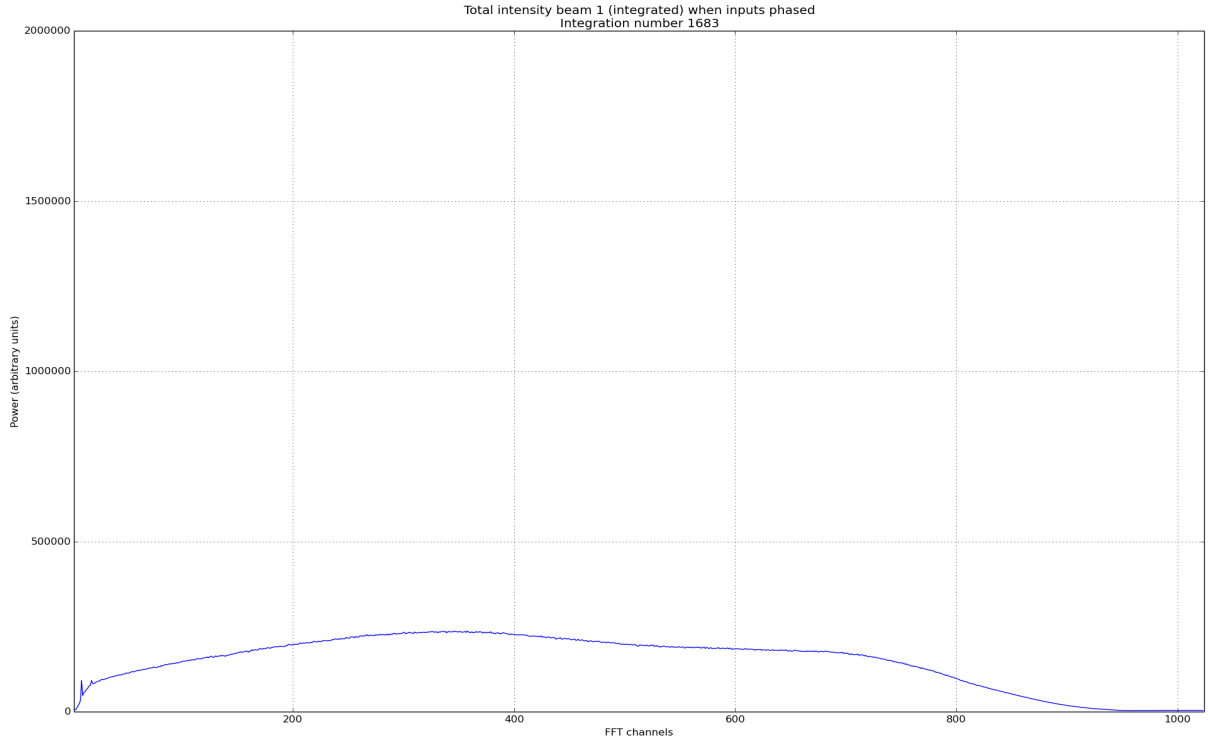


Fig.37 Beam band-shape of beam 2; beams can be observed in parallel

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

 # Testing of amplitude scaling for tapering:

(a) Amplitude scaling factor: 1



(b) Amplitude scaling factor: 2

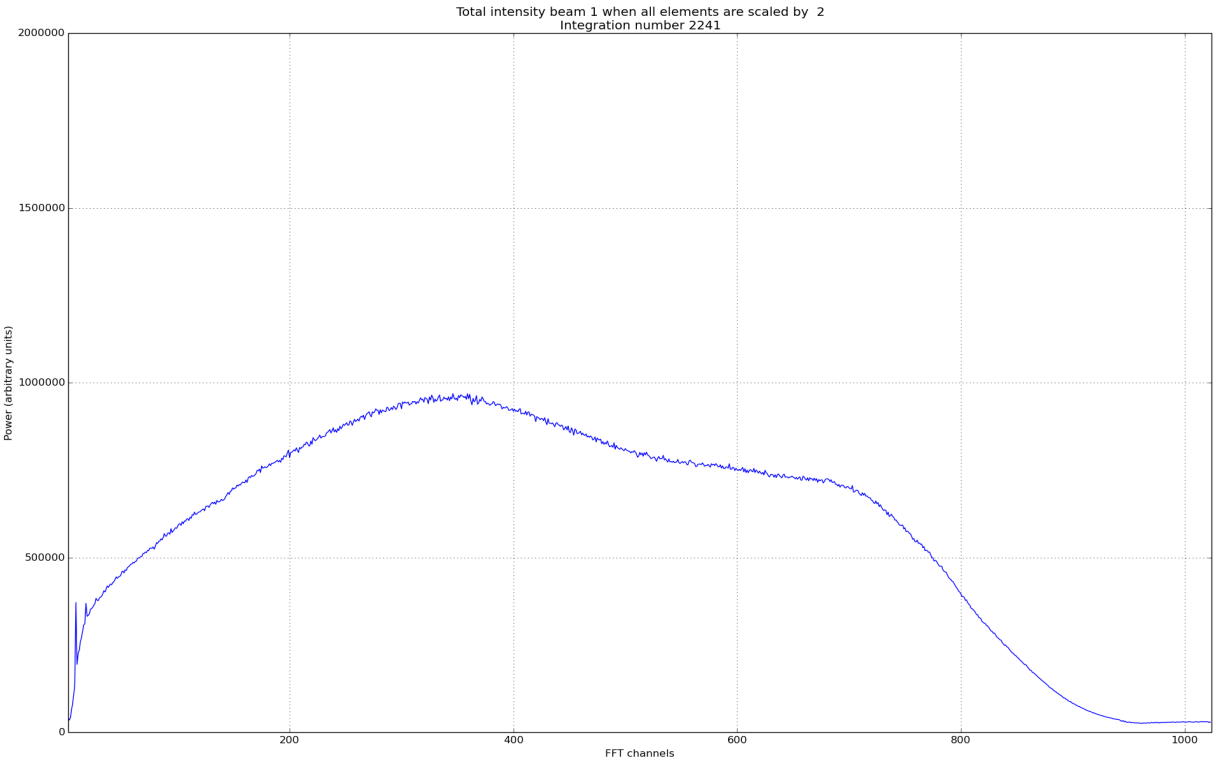


Fig.38 Effect of amplitude scaling 1 to 2; Y-axis range is same

(c) Amplitude scaling factor: 128

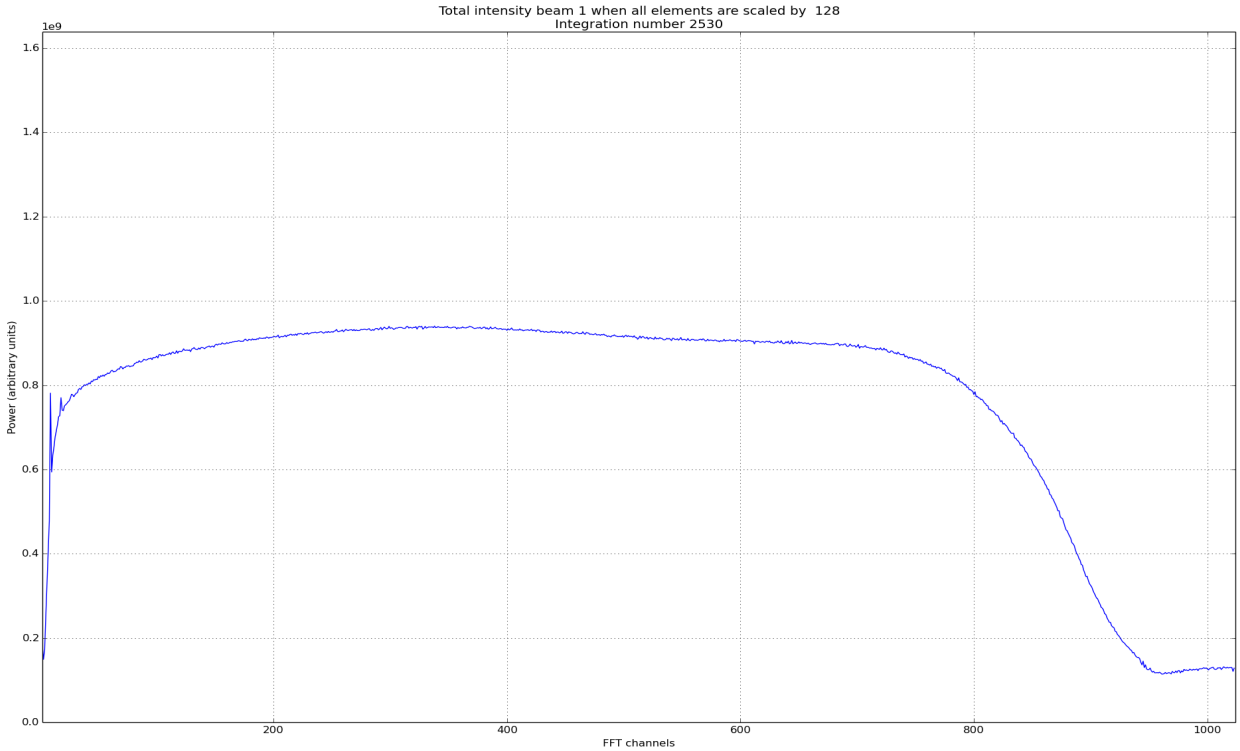


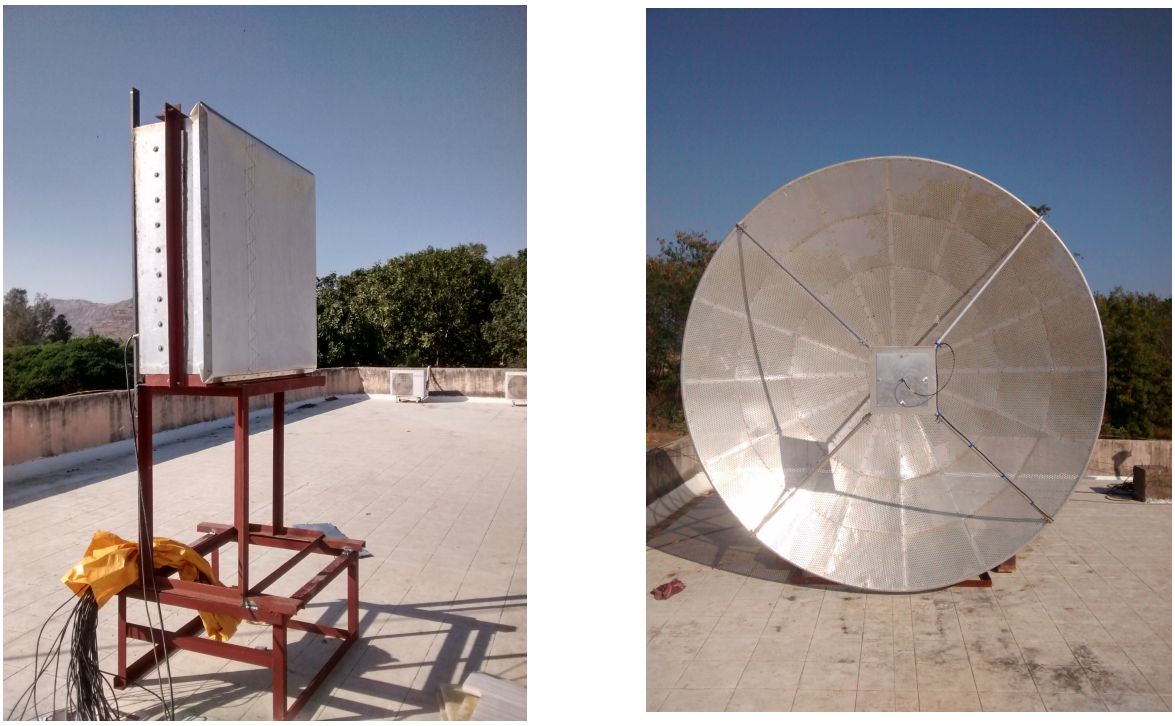
Fig.39 Effect of amplitude scaling by 254; saturation observed

 # Observations of testing with noise inputs:

- Broadband noise signal followed by 25 MHz LPF produces better band-shapes to observe correlation and beam results. (fig.32 to fig.39)
- With latest corrections in design, there is an improvement in sensitivity, so the signal has to be attenuated manually and/or digitally. Artefacts observed earlier in noise band-shape are not present now. (fig.32 to fig.39)
- When there is a difference in cable length, cross-correlation shows gradual phase along the band (fig.33(a)). These phases are corrected after phase multiplication. Fig.33(a) shows phasing of 'AL' pair of elements.
- Beam data shows power spectrum of summation (after FFT) of all 16 inputs, so doubling all inputs increases power 4 times as seen in fig.38.
- Depending upon input power level and digital beam gain provided, the range of Y-axis count values needs to be kept appropriate. Saturation is observed when all elements are scaled by higher scaling factors like 128, 255 as shown in fig.39 because saturation option enabled in vector accumulator.

Chapter 8. Tests with FPA

The FPA is being tested with free-space test range at GMRT. 3-m dish has been kept at ~67m away from the FPA. The arrangement is made in such a way that centers of both dish and FPA are aligned. This provides enough far field for L-band radiation tests as a starting point. Fig.40 shows both FPA(fig.a) and dish (fig.b) used for beamformer testing.



(a) (b)

Fig.40 (a) FPA (b) dish antenna for radiation

A. Basic tests

The latest version of 16-input, 4-beam narrowband beamformer has been tested with radiation from the 3m dish as shown in fig 40.(b). Carrier wave of 1.2GHz and 1.3 GHz was radiated initially. Dipole feed of dish was excited with different power levels through ~26m RG-223 cable. Power received at the output of feed elements as well as down-converting units (DCU) was observed. There is a lot of variation of power received by different feed elements. Also, the power received by one feed element varies over time around 1dBm. Causes of these variations are being debugged. Correct polarization was identified and CW signal power was measured at the output of down-converting unit (DCU). Typically, for -30dBm radiation provided through ~26m cable and received through ~10m at FPA side was observed at DCU output between -44dBm to -27dBm. This is the power which goes to ADC of beamformer set-up. This

was verified at the different times and was decided to use as appropriate signal power.

16-input, 4-beam narrowband beamformer has been tested where 16 elements of same polarization were chosen making a cross of vertical and horizontal elements. With this set-up, basic tests of correlation, phasing, beam steering have been carried keeping an element as a reference. Due to the fixed distance among elements, there was constant phase initially; this phase was corrected by phase multiplication successfully as shown in fig.41. Red plot shows cross-correlation phase of an element pair before phase correction and green plot (fig.41) represents cross-correlation phase after phasing. Initially, -50^0 was the phase between two elements which became $\sim 2^0$.

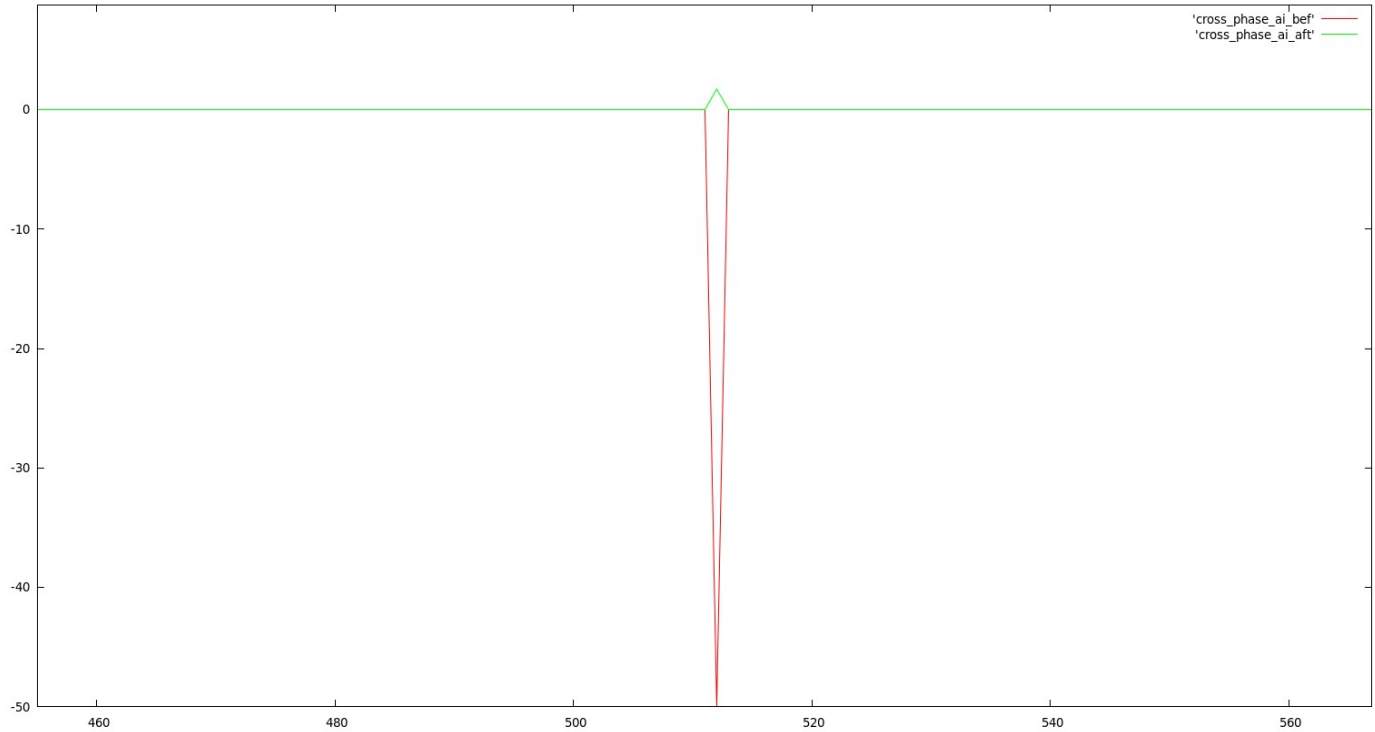


Fig.41 Change in cross-correlation phase spectrum after phasing

Variation of auto-correlation and cross-correlation magnitude was also observed while testing beamformer similar to sweep mode of the spectrum analyzer. Multiple beams can be observed simultaneously. Fig shows a plot of two beam band-shapes plotted at a time. Red plot of fig.42 shows beam 1 data without any steering and 8 elements in the phased state whereas green plot (fig.42) represents beam 2 data when it is steered by 4 degrees in the sky. Effect of addition of more number of

antenna elements was clear but variation of signal power and multiple path propagations makes somewhat difference in beam data.

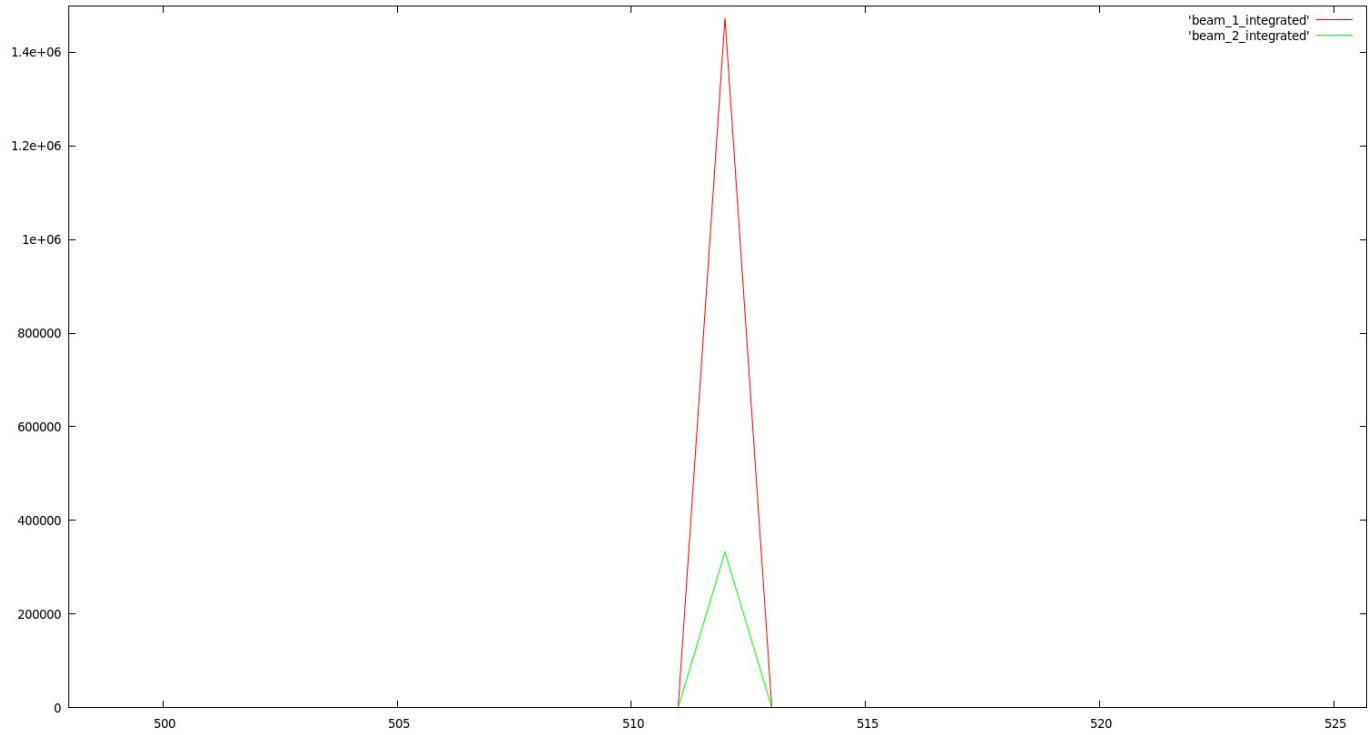


Fig.42 Observation of multiple beams, beam1 just phased, beam2 steered by 4^0

B. Beam steeering:

As horizontal row could provide a reflected signal from central building, vertical row of 8 elements was chosen and an element was kept as reference to get correlation phases. Remaining elements of beamformer were made as zero signals. Considering this as a uniform linear array of 8 antenna elements separated by 11cm, phase gradient was calculated for beam steering. Different angles of phase gradient were calculated to steer the up and down by 1 degree in the sky. Radiating dish provided narrow beam and vertical array makes the beam narrower. Bell-shaped curve of beam steering was observed as shown in fig.43. Half-power beam-width was observed to be ~13 degrees effectively. Count of CW signal power of a beam is plotted for each degree steering in the sky. This result appears to be acceptable considering approximation of free-space test-range.

Calculation for phase gradient:

$\Delta\varphi$ = phase shift between two successive elements

d = distance between the radiating elements; 11cm in this case

Θ_s = beam steering in sky

$$x = d \cdot \sin \Theta_s$$
$$360^0/\Delta\varphi = \lambda/x$$

$$\Delta\varphi = 360^{0.}. d. \sin \Theta_s / \lambda$$

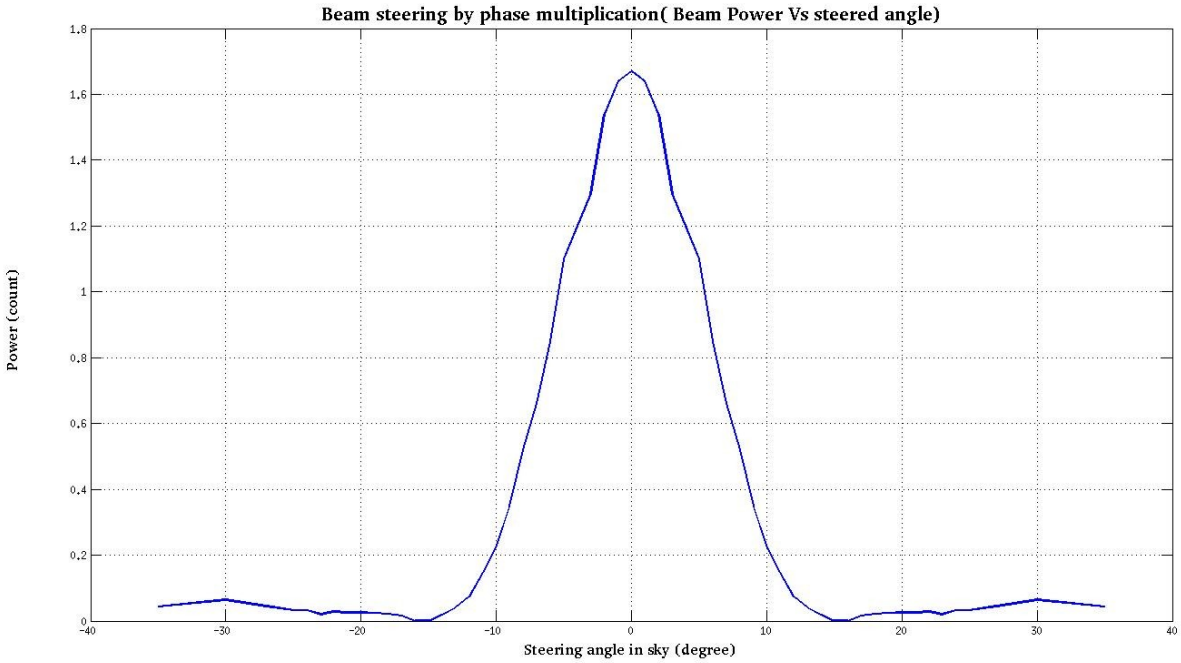


Fig.43 Vertical beam steering for a row of 8 for input CW 1.3GHz

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

C. IRNSS signal:

The narrowband beamformer has also been tested for a navigational satellite signal with FPA heading to the east direction. There are 3-4 IRNSS satellites which are present in the field of view of feed elements. Individual feeds have beam-width of ~120^0. So, there always some signal picked up by FPA. The frequency used by IRNSS is 1176.45MHz. This satellite signal was observed by proper local oscillator tuning. Fig.44 shows IRNSS signal when 16 feed elements were phased and RFI was not dominant in that band. There is a problem in beam steering of satellite signal because there are multiples satellite always present in FOV of feed elements. This creates confusion of change after steering of the beam. More experiments need to be done to achieve steering of sky source.

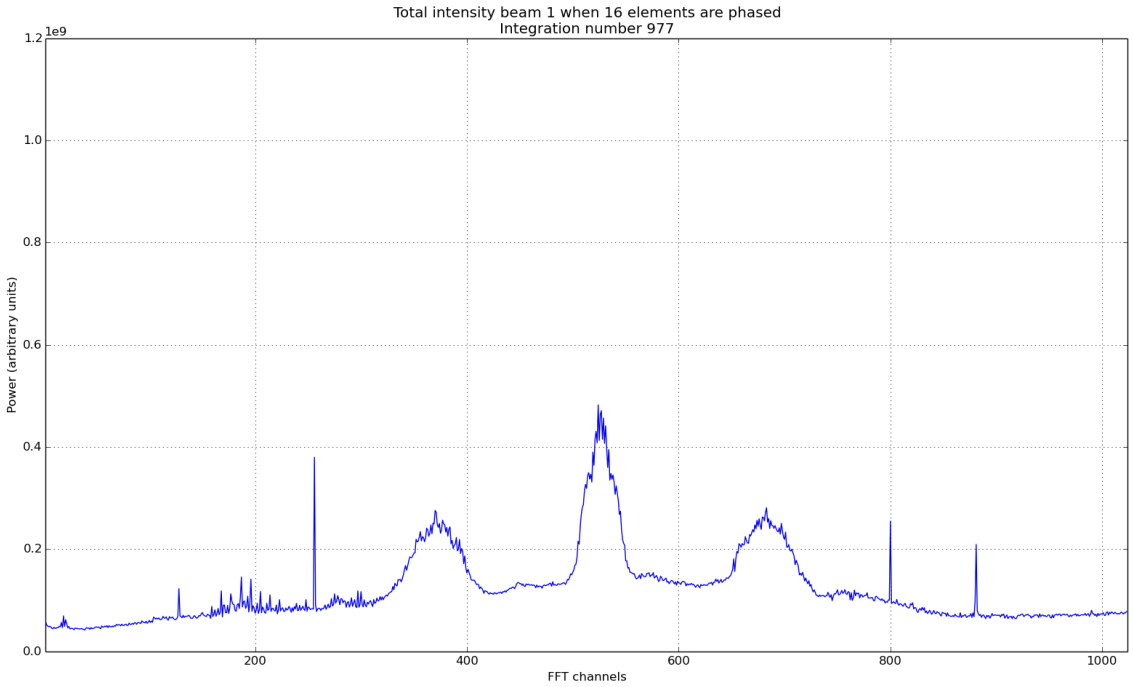


Fig.44 Total intensity beam of 16 phased feed elements and band set for IRNSS signal.

Chapter 9. Enhanced bandwidth of narrowband set-up

As this document describes 64-channel based narrowband FPA beamformer, there is additional feature achieved during development. 64-channel ADC has 50MHz onboard sampling clock and this limits the bandwidth of signals. There is provision of increasing sampling up to 65 Msps by connecting external clock to one of the ADC pin. The external clock has to be applied via differential lines and LVDS voltage levels. This option was being studied and discussed in CASPER community. Alternatively, one experiment was carried out in GMRT in which 50MHz oscillator IC was replaced by compatible IC (CWX813-064) successfully. This was tested with narrowband beamformer design applied with broadband noise source followed by low pass filter of 25 MHz and 30 MHz. Also, CW signal was varied in operating range till 32 MHz to observe proper working of enhanced bandwidth. Now we have 64-channel based narrowband set-up which can digitize, correlate and form beam up to 32 MHz.

Following fig.45 is sample result of cross-correlation with increased bandwidth:

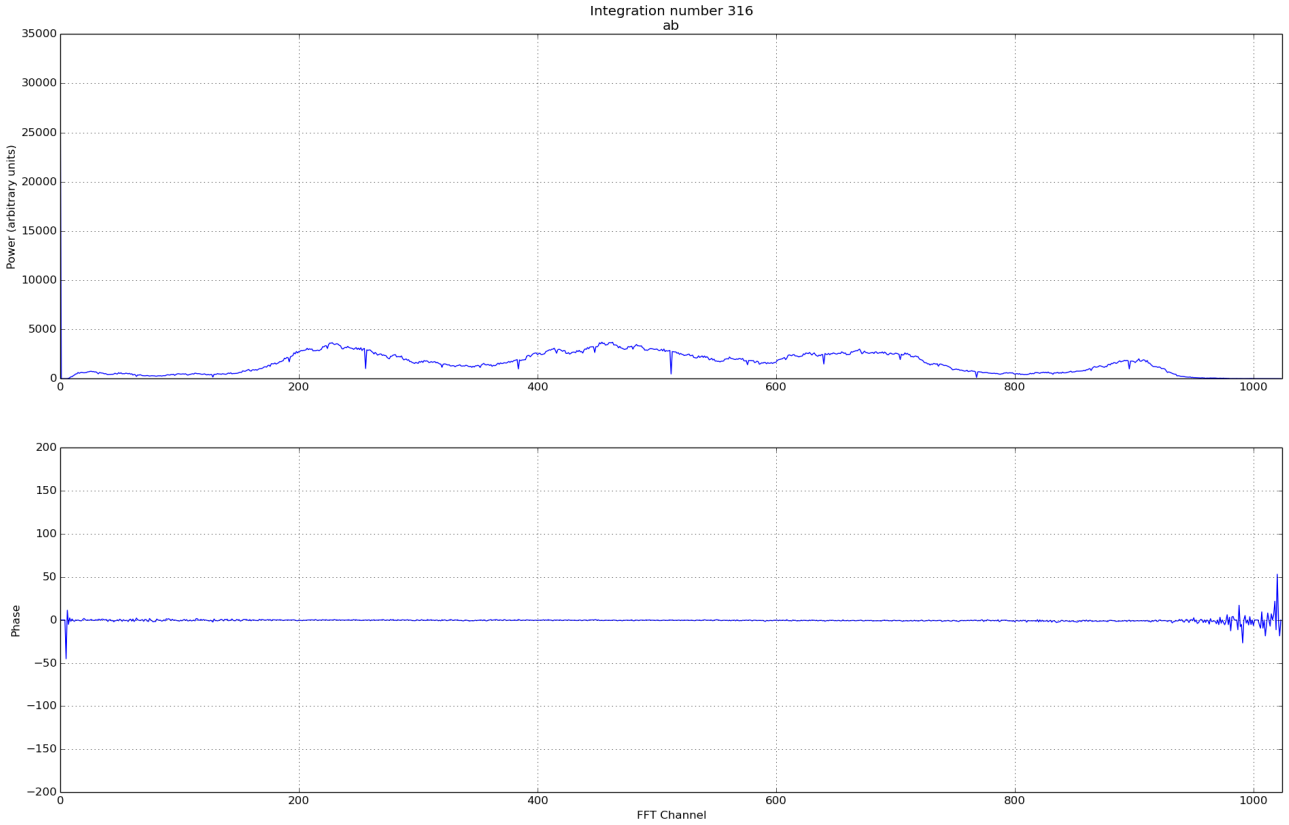


Fig.45 Cross-correlation observed after replacement with 64MHz clock;
Noise input given with 30 MHz LPF

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

Chapter 10. Enclosure for narrowband set-up

64-channel ADC has been interfaced to ROACH-1 board for narrowband set-up. Apart from enhancing bandwidth from 25MHz to 32MHz, an enclosure has been prepared for the set-up. Heating up of ADC chips was found when operated in digital back-end lab, so cooling arrangement has been added to set-up. Along with 3 in-built smaller fans in ROACH enclosure, two fans have been put with independent power supply. Metallic strip with heat-sink has been placed over chips to avoid heating up. These additional units goes beyond usual ROACH enclosure, therefore new enclosure has been prepared in GMRT as shown in fig.46

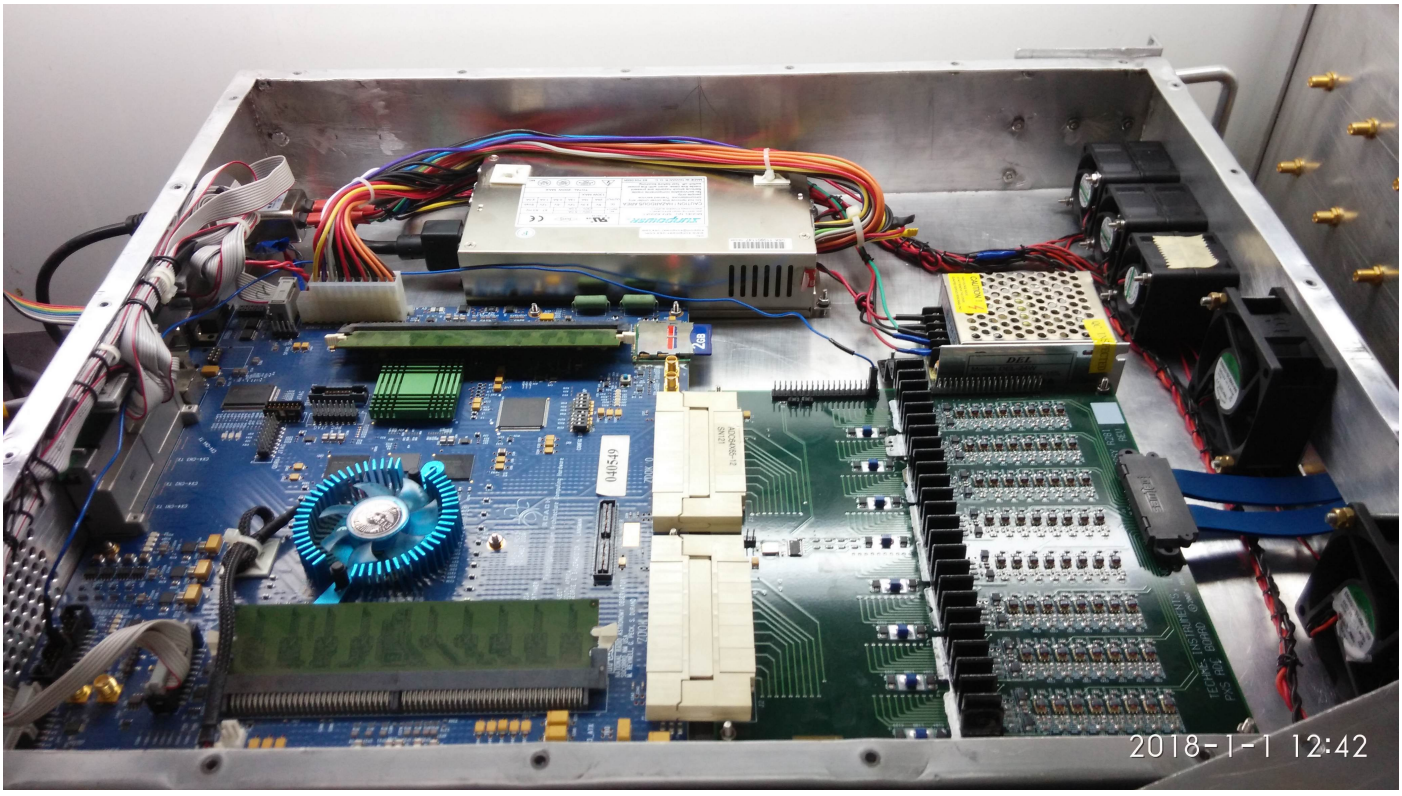


Fig.46 Enclosure of narrowband set-up after modification.

Chapter 11. Conclusions

The basic narrowband FPA beamformer has been developed. The basic steps of beamformer back-end for FPA has been identified for narrowband signals. 64-channel ADC interfaced to ROACH-1 board can serve the objective of FPA back-end development. Following are specific conclusions about development:

I. Data acquisition from 10 GbE of ROACH-1 has been checked on Wireshark. There isn't packet loss due to point to point connection between FPGA and host machine. There is limitation of writing data onto disk but continuous data grabbing by C code can to be tried further.

II. In case of the correlator, 15 pairs of cross-correlations and 16 computations of auto-correlations have been included in the design. Truncation used for correlation is 18 bit real/imaginary to 4 bits of real/imaginary. Available resource of FPGA can be used smartly to correlate as calibration required is not frequent. Approach to correlate 64-channel ADC data on FPGA has been identified and verified.

III. A basic model of coherent beamformer has been developed for 64-channel ADC data. The phase difference between the RF signals has been dumped and multiplied back to correct the phase and steer beams by a script. Amplitude scaling has been tested in the lab which is able to provide tapering up to 24dB between the FPA elements. Four beams are available in the latest design with control over amplitude and phase for each beam.

IV. Results of latest design mentioned in chapter 7 have been verified by an experiment of FPA test-range in GMRT (chapter 8). Now we have a design which can compute 2048 point FFT of 16 narrowband (32 MHz) baseband inputs. Cross-correlation of 15 pairs is used to phase all 16 inputs with one input as a reference. There are four beams whose complex weight can be applied independently. Phasing of

signals makes elements in phase which are summed up in beamforming resulting in a 16-bit value of total intensity.

VI. The study of FPGA resource utilization will be helpful to understand the tradeoff among dimensions of narrowband developments. It can be concluded that resources are required in either increment of beams, FFT size or number of inputs.

VII. The bandwidth of 64-channel based narrowband set-up has been successfully enhanced to 32 MHz by replacing oscillator IC on ADC board in GMRT. Working of 64Msps sampling has been tested with CW and noise source.

VIII. More testing is needed for current development to check finer corrections in design so that final procedure of implementation for FPA beamformer can be prepared.

Chapter 12. Future scope

1. Scaling of design:

Current design of 16-input, 4-beam is being tested for various cases and modifications. Once it is done, a finalized design would be scaled to more number of inputs of ADC like 32, 48, 64. For more number elements, two boards can be used. One board computing FFT and raw data capturing whereas other board for correlation/beam mode. A single board can serve initial purpose of 16 inputs or 32 inputs in two design modes.

2. FPGA Design pipeline:

In the current version of 16 input design, correlation is stored in BRAM & being read by a python script. For actual implementation of the beamformer, beam data has to be recorded continuously. Data would be carried on 10GbE port of ROACH. This can be captured by a machine for further processing.

3. Addition of Polyphase Filter Bank:

Introducing polyphase filter bank with appropriate windowing can improve performance by reducing scalloping loss. For slower data of 64-channel ADC, it can be difficult to add PFB-FIR bank. It can be achieved by software processing or creating FPGA design block for slower operation. This possibility can be explored.

3. Software processing:

Narrowband packetizer can send digitized data to a machine continuously. This raw voltage can be used for software processing of correlation and beamformer. The code used for 15-m dish back-end is being tested for continuous data grabbing into a computing machine.

4.Testing with FPA set-up:

The design has been tested inwith FPA once. This would be tested with focal plane array with different test cases again.

5. ROACH-2 environment:

All the development is being done on the ROACH-1 environment. ROACH-2 has much more resources than ROACH-1, hence design can be scaled easily. The ROACH-2 option will be tried as a second board for narrowband beamformer development

Chapter 13. References:

1. CASPER wiki pages have been extensively referred to many aspects of development.

a. https://casper.berkeley.edu/wiki/ROACH
b. https://casper.berkeley.edu/wiki/Tutorials
c. https://casper.berkeley.edu/wiki/Toolflow
d. https://casper.berkeley.edu/wiki/X64_adc
e. https://casper.berkeley.edu/wiki/Fft_biplex_real_2x
f. https://casper.berkeley.edu/wiki/ROACH_NFS_guide
g. https://casper.berkeley.edu/wiki/Block_Documentation

2. Oxford beamformer & correlation design of an application using 64-channel ADC.
3. LOFAR documents for understanding FPA set-up CEP data recording

Appendix A

CEP data recording [ref.3]

LOFAR system from ASTRON has DCU, RCU, RSP units along with Focal Plane Array(FPA) feed. FPA data is down-converted by DCU unit and given to receiver unit(RCU). RSP boards process beamforming controlled by control machine (LCU). There are two ports from RSP boards i. e. control and data. Control port gives integrated beam data whereas data port gives non-integrated data for raw data recording to the central processor (CEP). Instructions were followed to prepare CEP data recording as per document received from ASTRON. Pelican-Lofar software on a machine is used to record raw data. As the instructions and commands for installations are not compatible with current OS versions, only one software could be installed out of two required software. So this CEP data recording from the data port of RSP couldn't be proceed. This is a pending issue due to nonsupported commands and operating system.

Appendix B

Server Configuration:

Newly purchased HP Z640 server has been set-up for e-GMRT activities. Ubuntu 14.04 has been installed and configured the system for MATLAB compilations & ROACH booting environment. The server machine has been assigned 192.168.5.46 IP in digital back-end lab. MSSGE setup with Xilinx 14.x and MATLAB 2012b has been configured for CASPER development [ref. 1.c]. This environment can be used for ROACH-1 and ROACH-2 compilations. ROACH NFS guide [ref. 1.f] has been referred for networking based booting of ROACH. Python packages have been installed on the server machine. The server was configured for ROACH-2 booting but now it is being used as ROACH-1 boot environment. Driver for Myricom 10Gb Ethernet card has been installed. Gulp utility has been added and tested on the server for 10Gb data from ROACH.

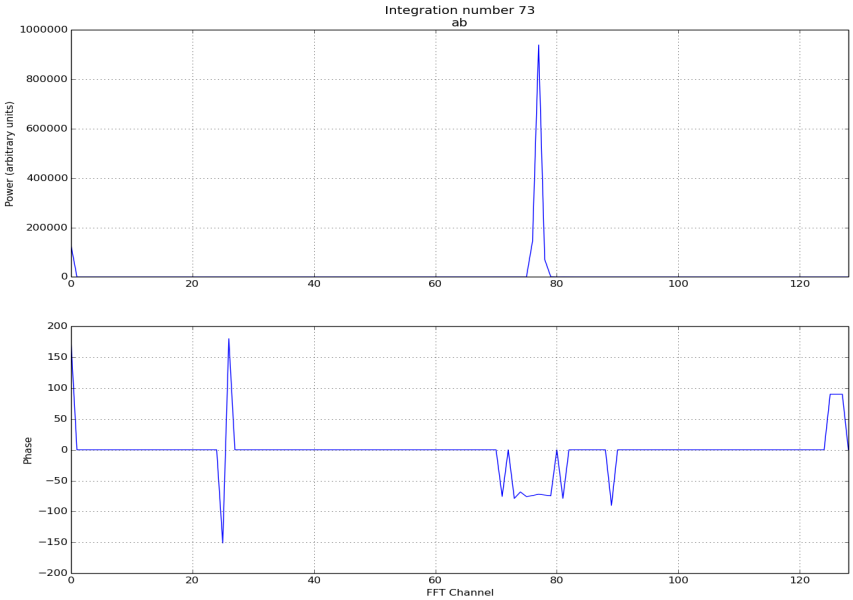
Appendix C – Intermediate designs

- ***8-element experiment:***

i. CW input-

- CW 15MHz (-27dBm) is given through power divider as two ADC inputs.
- Fig.47 shows phase before phase correction and after phase correction

(a)Before:



(b)After:

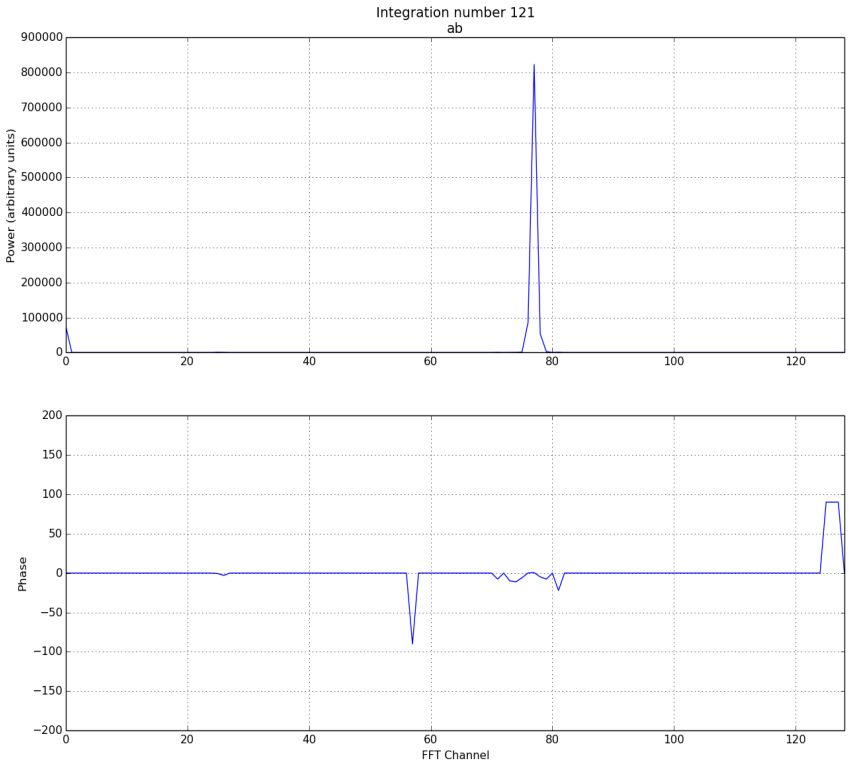
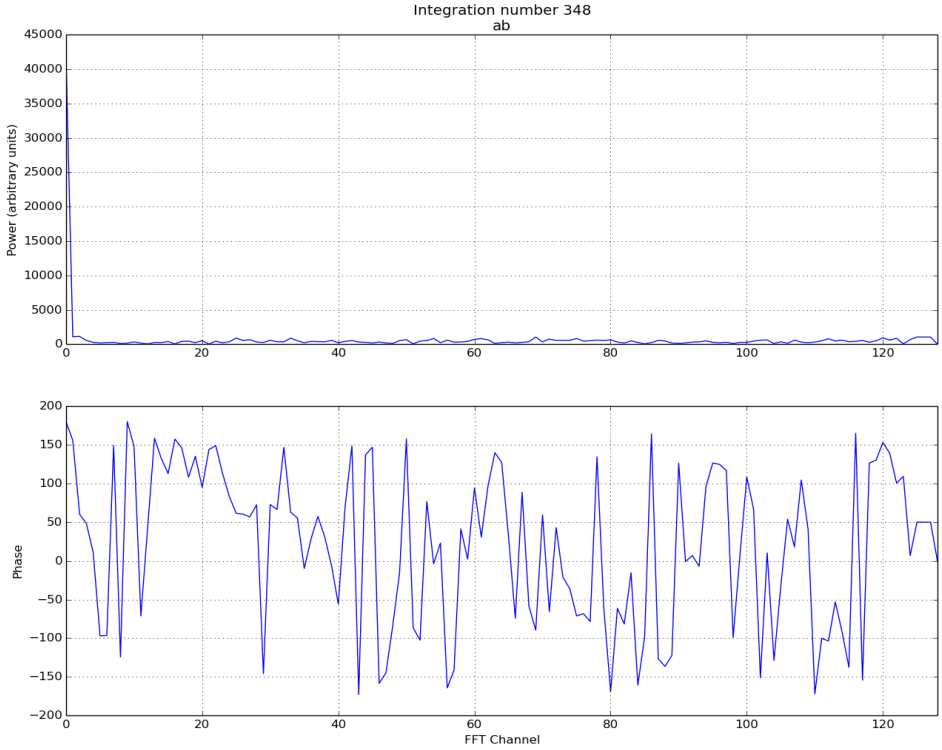


Fig.47 Phasing of CW signals

ii. Noise input-

- Broadband noise generator is used with 25MHz low pass filter and power divider
- As two noise inputs were uncorrelated, we can see phase variation and no correlated power in the magnitude spectrum (refer fig.48)

(a)Before:



(b)After:

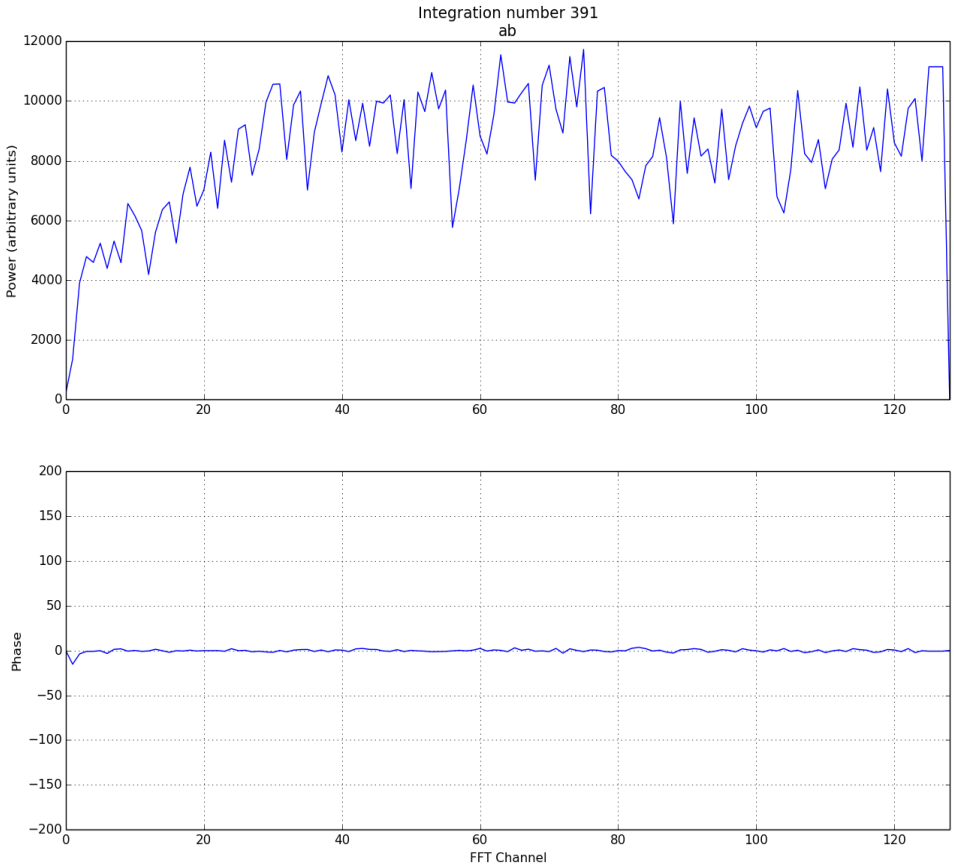


Fig.48 Phasing of noise signals

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

#Addition & cancellation for noise:

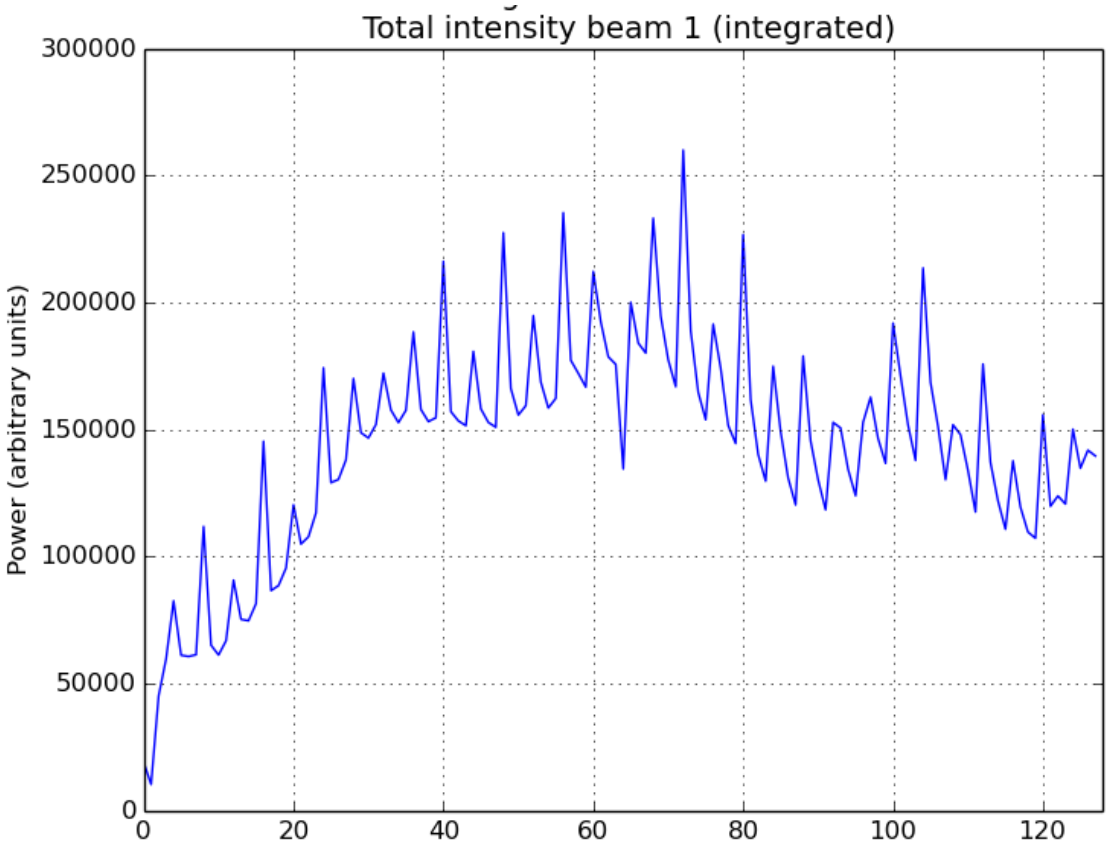


Fig.49 Addition of two correlated noise signals

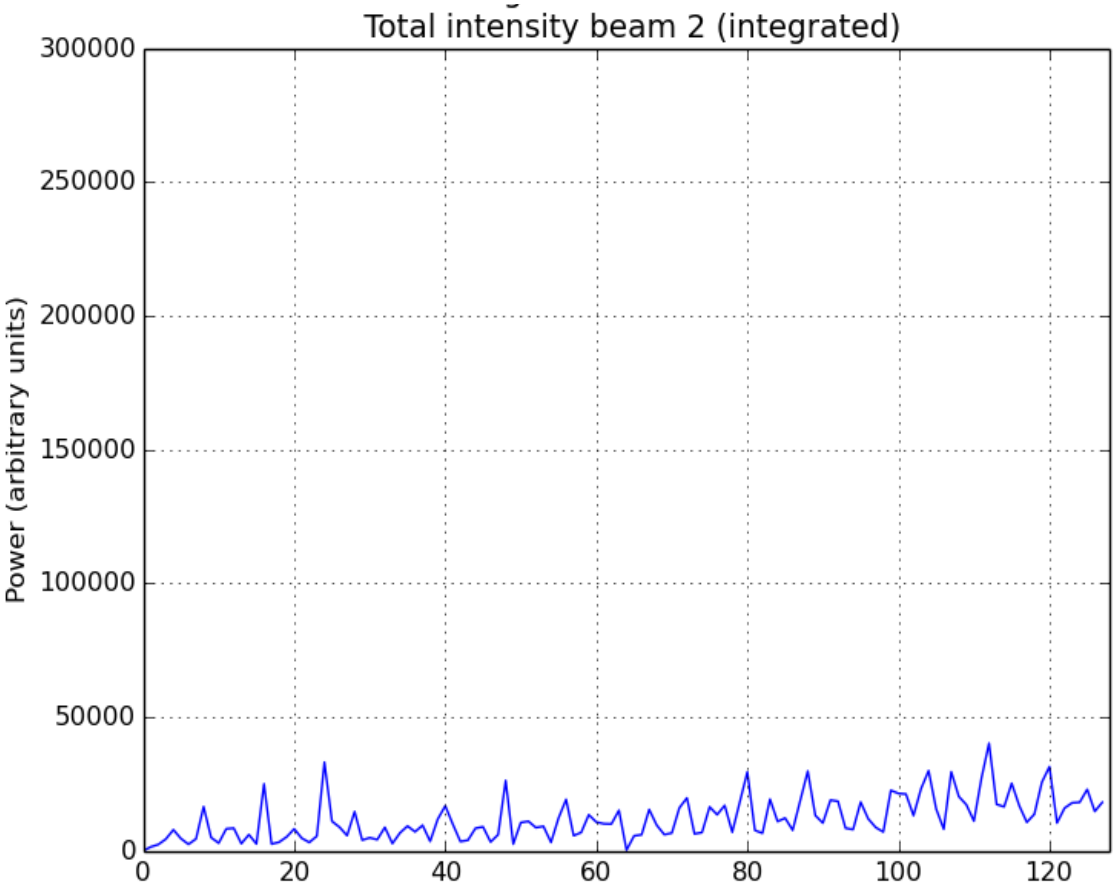


Fig.50 Addition of uncorrelated noise signals (Cancellation)

#Analysis of results:

Fig.27 to fig.50 are the initial results of narrowband beamformer developed using 64-channel ADC. Phasing of CW & noise, correlation stability, addition & cancellation, the addition of antenna in the beam, sample test of amplitude scaling etc. were tested. Results were discussed and found following conclusions:

1. Phasing part in both cases of CW and noise works well.
2. Correlation shows stability over time but needs to be tested with multiple channels of noise and for a longer time.
3. There is an unexpected phase of cross-correlation between some of the signals. This is causing ripples in beam and correlation of noise.
4. Truncations of bits can be changed as 36 bits to 16 bits for correlation and beam should be tried for next level designs.

#Erratic phase in correlation:

After testing this problem design in many ways, it was found that two of serialized signals are showing some delay in the whole signal chain. Except first 2, all other separated antenna elements in serialized FFT stream show in-phase. So beam was observed with flagging first 2 elements, it showed good band-shape. This option was explored more with 12-element beamformer design.

- ***12-element design:***

The experiment of excluding suspected antenna elements were ignored and 12 input design was formed. This design considers 6 elements of the first serialized stream and 6 elements of second serialized stream. 11 cross-correlation pairs are calculated, 2 beams are achieved with amplitude and phase correction. 8 bits are used in amplitude scaling, 2K LUT values for 360-degree phase resolution. Some of the elements showed some ramp in phase spectrum of cross-correlation which gets corrected after phasing. This phase is consistent with beam result, could be observed due to finite delay in elements. Band-shape of beam result is without ripples and amplitude scaling smooths beam band-shape more because of truncation. Results of 12 input design are shown in fig.51 to fig.54:

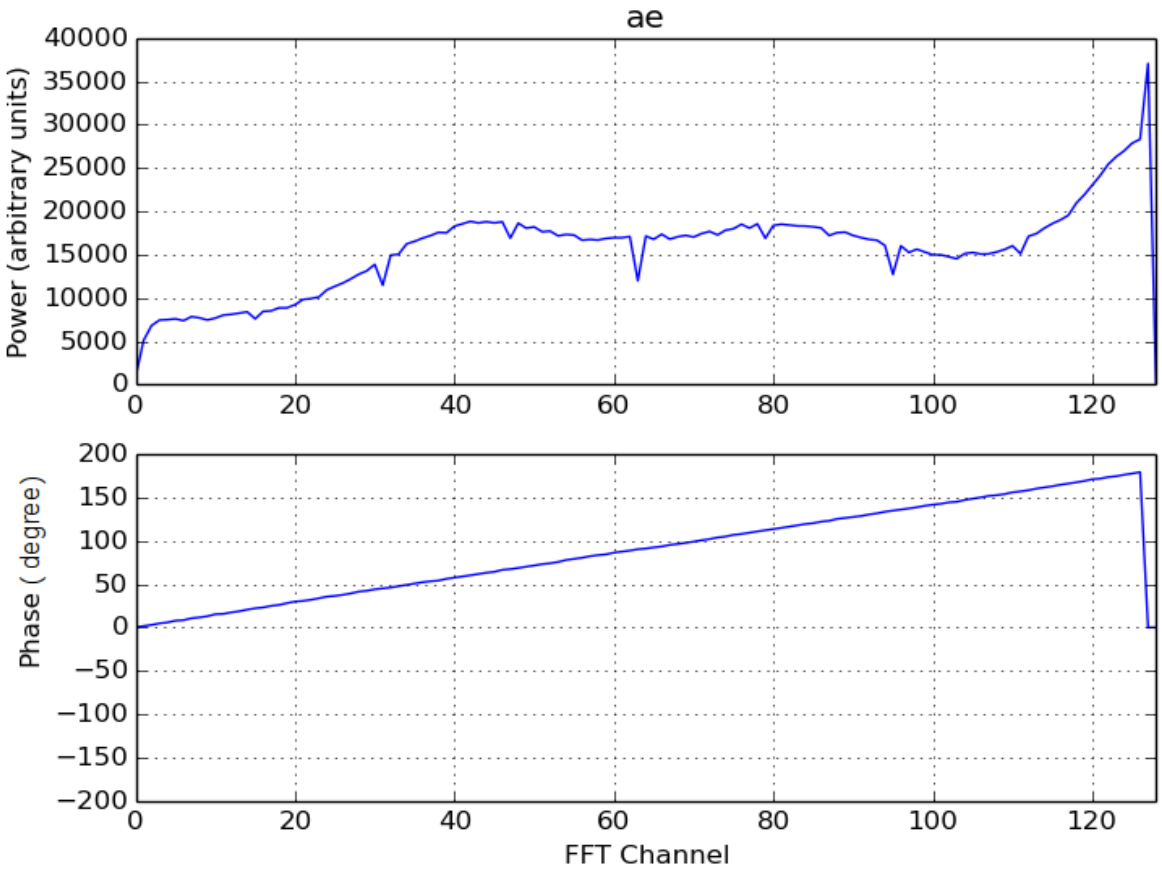


Fig.51 Cross-correlation of one of the pair of 12 input design

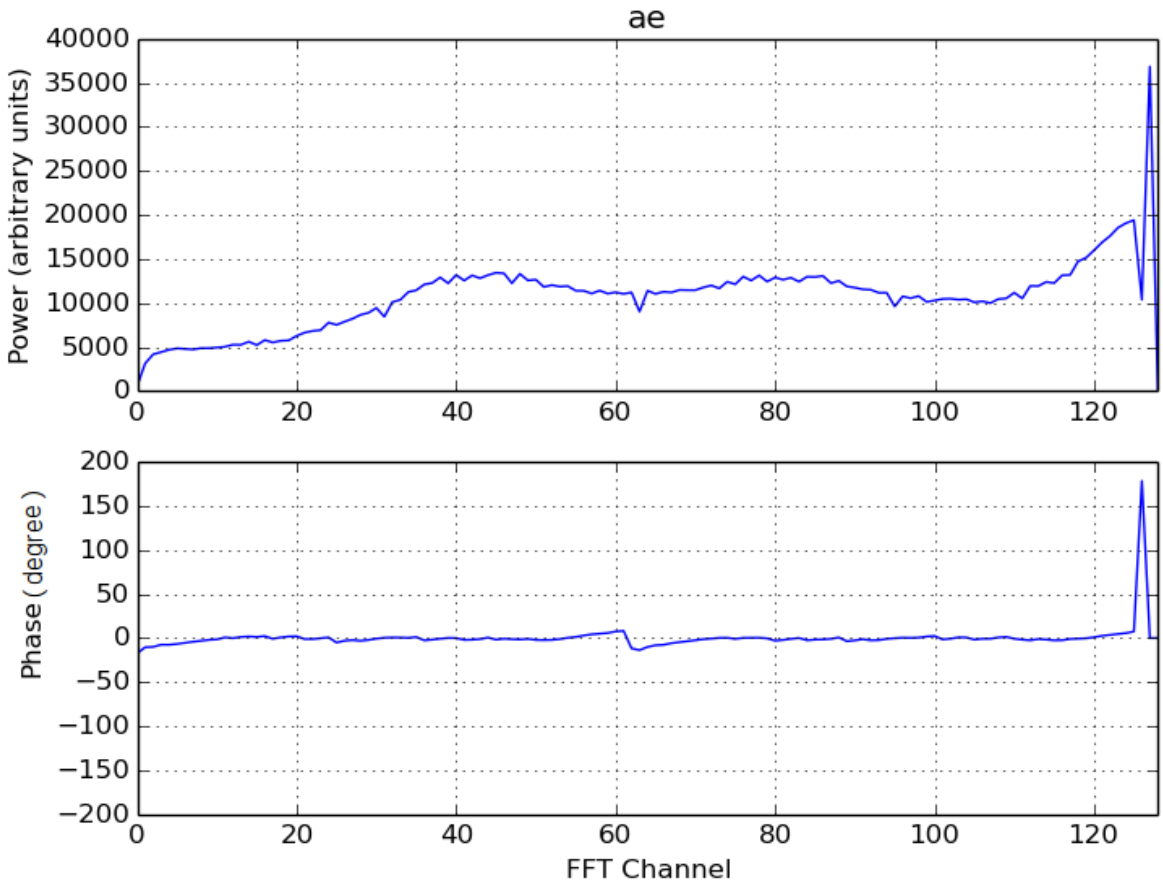


Fig.52 Phasing of above pair of 12 input design

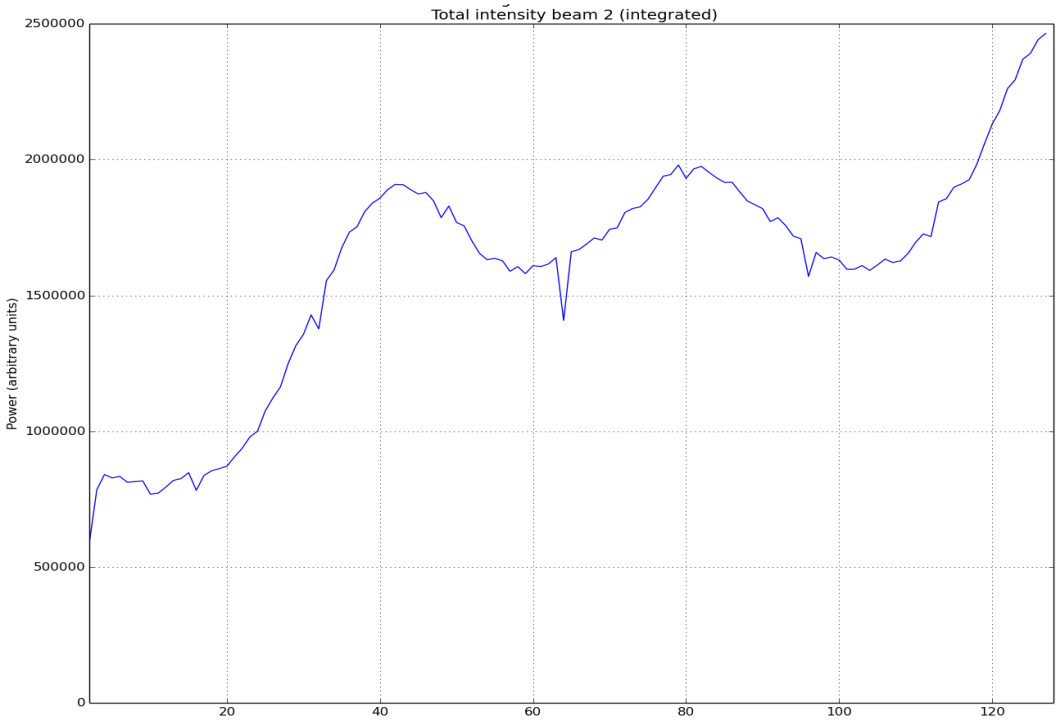


Fig.53 Beam band-shape of 12-input design after phasing all elements

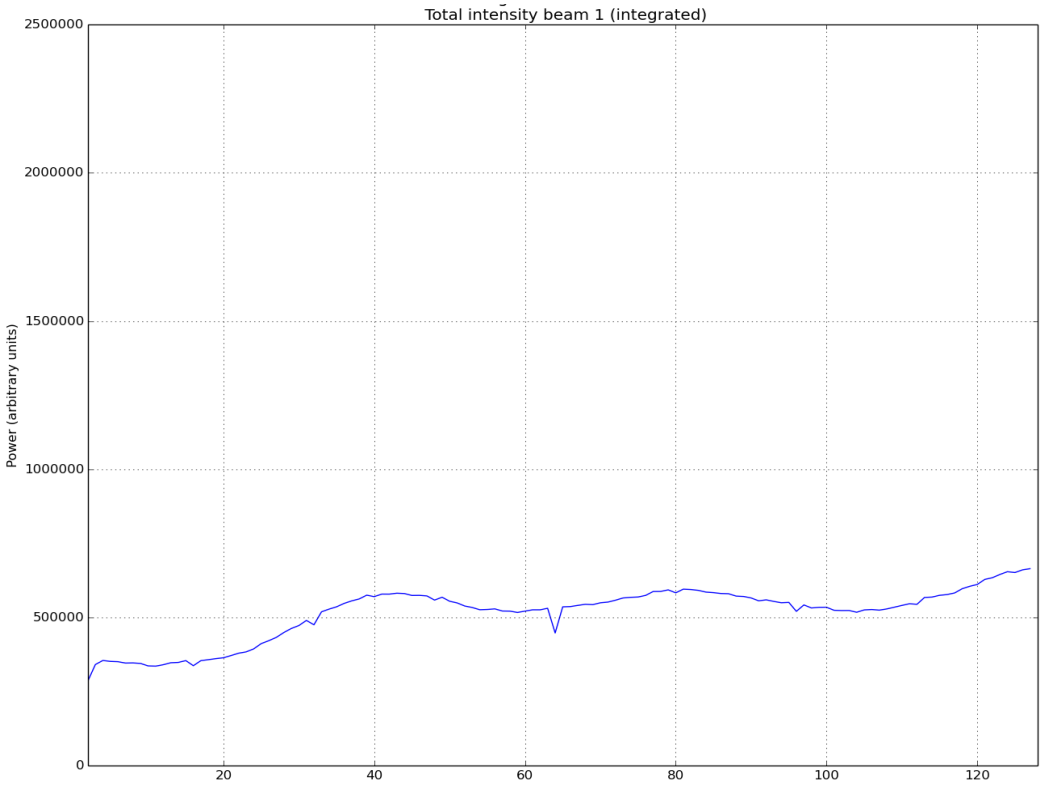


Fig.54 Beam band-shape of 12-input design after amplitude scaling

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

#Solutions of erratic phase in correlation:

There is some delay in first 1-2 signals of the stream of 8 serialized FFT outputs. This was observed in the correlation of 8 input design. Many ways were tried to know the probable reason and specific point in design. As all the design logic is dumped in an FPGA chip, one can simulate, make a change in the design and observe the expected result. On debugging the design flow at multiple points, it was found that there are 3 types of cross-correlation phases for antenna signals.

a. random phase along the band

b. multiple ramps across the frequency band-shape

c. single ramp across the band

First type of delay was unusual and observed specially with the first signal of FFT output. This delay could be the outcome of tough practical timing considerations of FPGA. It was found that it doesn't appear in a signal if it is delayed compared to other elements. So a design has been prepared with delayed lines for initial two signals taking care of proper delays in signal flows. Use of two additional FIFO has solved the problem of total uncorrelated phase. Second & third types of the delays observed can be caused due to fixed delay in the signal flow. The second type of delay causing multiple ramps in correlation was observed because initial state of one FIFO in time domain was ignoring some samples of data. The third type of single ramp in few correlations was solved by a delay of a clock cycle. Flexible delays have been placed in some of the places in design.

Appendix D – Packet format and channel mapping

Narrowband packetizer design described in chapter 3 uses UDP packets explained in fig.55 and fig.56. Port-wise packet structure is shown in figure. 16 channels of 64 channel ADC are grouped as mentioned in chapter 3. Apart from 42 bytes of UDP header, 8 bytes of packet number is added at the start of frame. 16 bytes of each channel repeats for 512th times to make 8192 bytes of data. Channels are numbered in design interface. These are mapped from connection box as shown in fig.57.

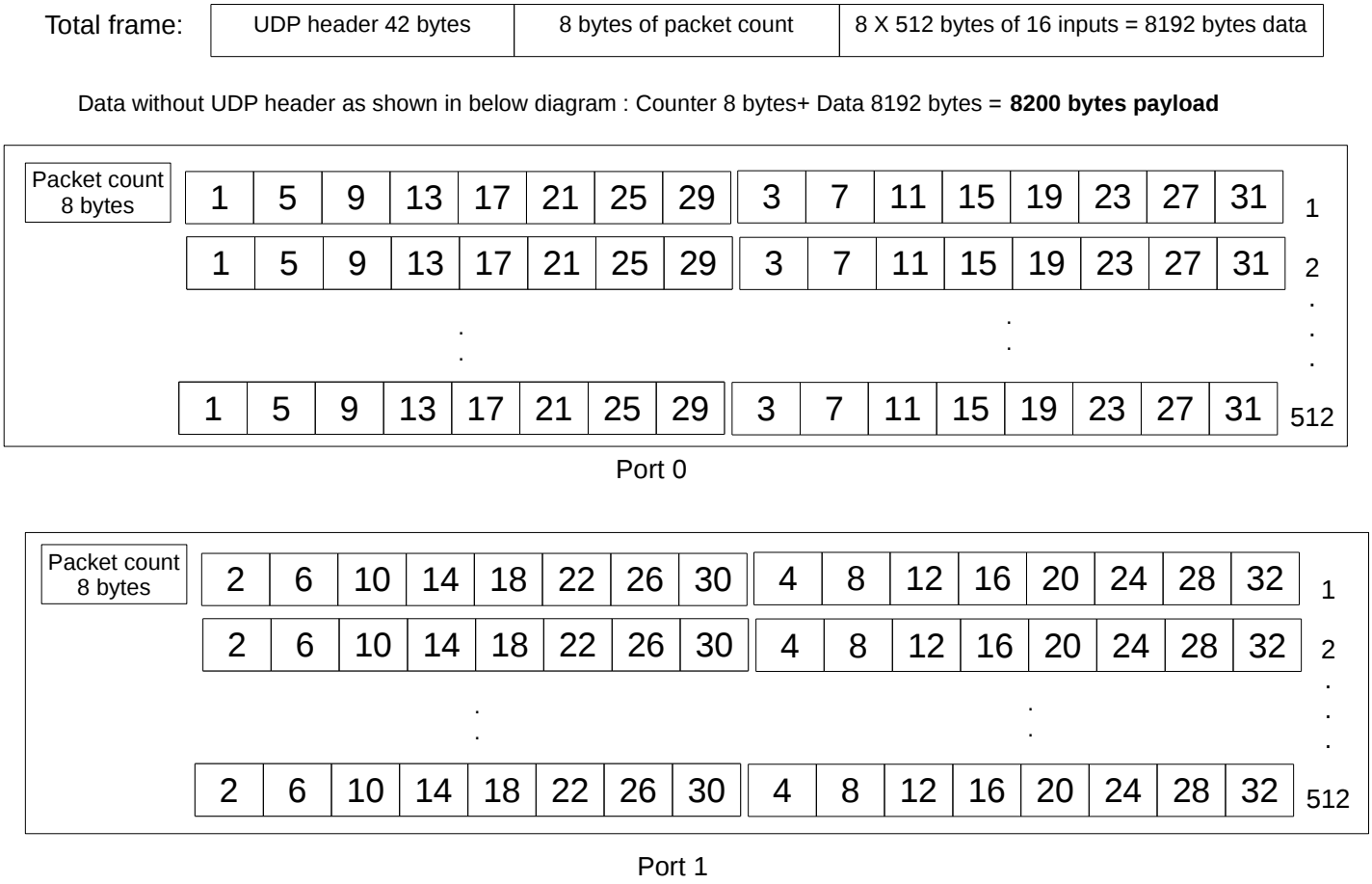


Fig.55 Port wise ADC data sent over an ethernet packet (port 0 and port 1)

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

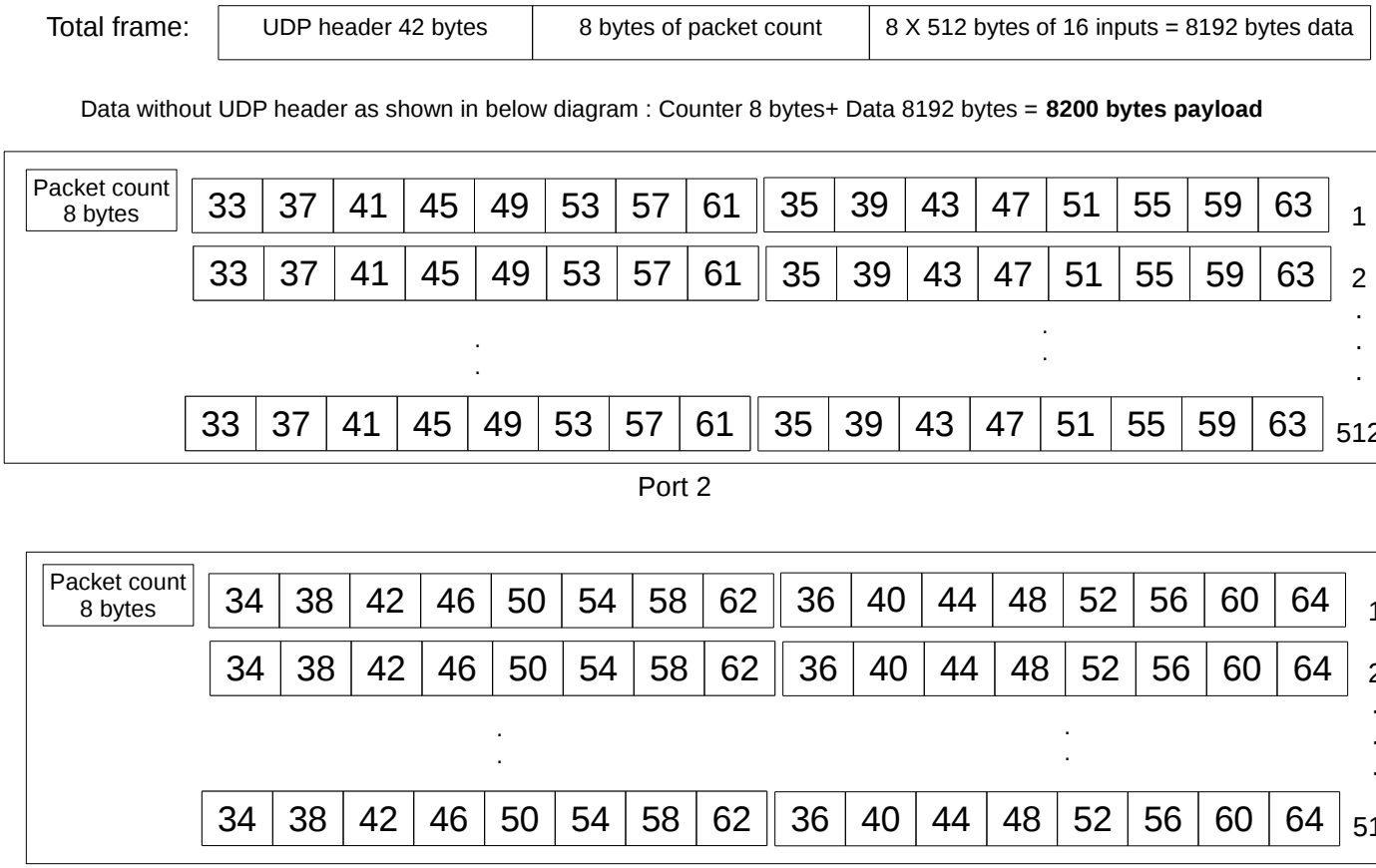


Fig.56 Port wise ADC data sent over an ethernet packet (port 2 and port 3)

Connection box mapping for 64 channels:

Fig.57 Connection box mapping for 64 channels to ADC-ROACH

Top

40	36	39	35	38	34	37	33	IC 5
32	28	31	27	30	26	29	25	IC 4
24	20	23	19	22	18	21	17	IC 3
16	12	15	11	14	10	13	9	IC 2
8	4	7	3	6	2	5	1	IC 1

Bottom

Conection box : 1 – 40 input side

Right side

Top

57	61	58	62	59	63	60	64	IC 8
49	53	50	54	51	55	52	56	IC 7
41	45	42	46	43	47	44	48	IC 6

Bottom

Left side

Conection box : 41 – 80 input side

Revision	Date	Modification/ Change
Ver. 2	11 May 2018	Corrected after first review

Appendix E – Channel mapping for beamfomer

Development of 16-element narrowband FPA beamformer is described in chapters 4, 5, 6, 7. There were different ways followed throughout the development. Chapter 7 explains latest version of 16-element design. This appendix-E provides mapping of ADC channels and final naming used in plots. 16 ADC channels are received in FPGA by 4 logical connections (dout0 to dout3). Assume signals A to P are received from ADC and reordered and sent to FFT computation. After FFT, they are shuffled and renamed for the convenience of plots.

#Output of x64_adc interface in FPGA-

Dout0 : A B C D

Dout1 : E F G H

Dout2 : I J K L

Dout3 : M N O P

#After time delay of 1 clock cycle before reorder-

Dout0 : D A B C

Dout1 : H E F G

Dout2 : L I J K

Dout3 : P M N O

#Output of FFT_biplex_2x-

pol02_out: D L A I B J C K

pol13_out: H P E M F N G O

#Renamed output of FFT_biplex_2x-

pol02_out: D L A I B J C K ==>> **A B C D E F G H**

pol13_out: H P E M F N G O ==>> **I J K L M N O P**

This is the final naming used for correlation and beamformer and final signal names are written on connection box. As per channel numbering used in fig.57, this sequence can be written as:

A B C D E F G H ==>> 4, 12, 1, 9, 2, 10, 3, 11

I J K L M N O P ==>> 8, 16, 5, 13, 6, 14, 7, 15.

Acknowledgement

We would like to express sincere thanks to Prof. Jayaram Chengalur, Prof. Nissim Kanekar, Prof. Yashwant Gupta for guidance and support throughout the development. We thank all e-GMRT team members for valuable discussions, feedback during monthly meetings. We thank to staff members from Analog and Digital back-end lab for their support and co-operation. Also, we thank staff members from Mechanical department for their help to prepare experiment's setup.