VENTSPILS AUGSTSKOLA

STATE RESEARCH PROGRAMME

“The Next Generation of Information and Communication Technology Systems”

„Nākamās paaudzes informācijas un komunikāciju tehnoloģiju (IKT) pētniecības valsts programma (NexIT)”

Projekts Nr.4

"Jaunas paaudzes liela apjoma datu apstrādes sistēmas"

1. posms

Beamforming Algorithm Realization Principles
(Stara formēšanas algoritma realizācijas varianti)

Author:	Evita Vaviļina
Report period:	1
Work package, number (WP):	4
Document version number:	2.1
Submission date:	30.03.2015.

Table of Contents

Introduction

Radio astronomy is a fast changing science, more importantly in the past few years it has changed a lot and become more advanced. Most of these changes are relative to the instruments of radio astronomy itself, i.e. radio telescopes. The classic radio telescopes are big dish directional antennas and all the experiments are performed with hardware sensors connected to those antennas. Unfortunately, there are engineering limitations in the dimension that a single dish telescope can achieve and furthermore the building of special purpose hardware instruments that are increasingly complex is expensive and does not offer flexibility.

Radio interferometry gives engineers the possibility of building bigger radio telescopes, instead of using a large single dish antenna; it combines the signals received from many different antennas to form a virtual radio telescope that is the combination of all the small antennas. At the beginning this technique was used to connect only antennas close to each other, but nowadays it is possible to connect antennas that are thousands of kilometres away, and so it is now possible to obtain radio telescopes with apertures so big that were not even imaginable ten years ago. The LOw Frequency ARray (LOFAR) radio telescope is an example of this new generation of telescopes.

In this context the beamforming algorithm and the subject of this report, acquires an increased importance. This technique, in fact, permits to combine different signals, received from many antennas, and to form a single coherent signal, called a beam. Moreover, beamforming permits to give directionality to an array of non-directional antennas.

Regarding the facts mentioned before, this report includes definition of beamforming, basic information about beamforming and an overview of beamforming algorithm realization methods.

1. Beamforming Basics

Beamforming or spatial filtering is a signal processing technique used in sensor arrays for directional signal transmission or reception, in other words, to control the directionality of sensor array. In a phased antenna array this is achieved by combining elements in a way that signals at particular angles experience constructive interference while others experience destructive interference. Beamforming can be used at both the transmitting and receiving ends in order to achieve spatial selectivity. In this work, we focus only on using beamforming for reception, i.e., to combine the signals received from an array of antennas and simulate a larger directional antenna. The improvement compared with omnidirectional reception/transmission is known as the receive/transmit gain.

Beamforming can be used for radio or sound waves. It has found numerous applications in radar, sonar, radio astronomy, seismology, wireless communications, acoustics, and biomedicine. Adaptive beamforming is used to detect and estimate the signal-of-interest at the output of a sensor array by means of optimal (e.g., least-squares) spatial filtering and interference rejection.

The problem when combining signals received from different antennas is that the receivers are in different places in space, and so each of them is receiving the same signal emitted by a given source at different time. Simply combining the signals received by the different antennas does not produce meaningful information, because the waves are interfering. But these interferences can be both constructive and destructive, and exploiting the behaviour of constructive interfering waves is exactly what beamforming is based on. The simplest beam former can be built just by connecting nearby antennas to the same receiver with wires of different lengths, thus delaying the signals and producing a temporal shift and an increase of sensitivity on a specific direction. This solution is not very flexible and beam formers are actually implemented with special purpose hardware, or with software.

In general, to form a beam from different received signals, a different complex weight is multiplied with each signal and then all the signals are summed together. The complex weight depends on the source of interest’s location and the spatial position of the antennas.

2. Beamforming Techniques

The beamforming technique used in receiving involves combining delayed signals from each antenna at slightly different time (the antenna closest to the target will be combined after the longest delay), so that every signal reaches the output at exactly the same time, making one loud signal, as if the signal came from a single, very sensitive receiver.

With narrow-band systems the time delay is equivalent to a "phase shift", so in this case the array of antennas, each one shifted a slightly different amount, is called a phased array.

In the receive beam former the signal from each antenna may be amplified by a different “weight”. Different weighting patterns (e.g., Dolph-Chebyshev) can be used to achieve the desired sensitivity patterns. A main lobe is produced together with nulls and side lobes. As well as controlling the main lobe width (the beam) and the side lobe levels, the position of a null can be controlled. This is useful to ignore noise or jammers in one particular direction, while listening for events in other directions. A similar result can be obtained on transmission. The topic of mathematics on directing beams using amplitude and phase shifts is covered in the third chapter.

Beamforming techniques can be broadly divided into two categories:

- Conventional (fixed or switched beam) beam formers;
- Adaptive beam formers or phased array:
  - Desired signal maximization mode;
  - Interference signal minimization or cancellation mode.

Conventional beam formers use a fixed set of weightings and time-delays (or “phasings”) to combine the signals from the sensors in the array, primarily using only information about the location of the sources in space and the wave directions of interest. In contrast, adaptive beamforming techniques generally combine this information with properties of the signals actually received by the array, typically to improve rejection of unwanted signals from other directions. This process may be carried out in either the time or the frequency domain.

As the name indicates, an adaptive beam former is able to automatically adapt its response to different situations. Some criterion has to be set up to allow the adaption to proceed such as minimising the total noise output. Because of the variation of noise with frequency, in wide band systems it may be desirable to carry out the process in the frequency domain.

Beamforming can be computationally intensive. Sonar phased array has a data rate low enough that it can be processed in real-time in software, which is flexible enough to transmit and/or receive in several directions at once. In contrast, radar phased array has a data rate so high that it usually requires dedicated hardware processing, which is hard-wired to transmit and/or receive in only one direction at a time. However, newer field programmable gate arrays (FPGA) are fast enough to handle radar data in real-time, and can be quickly re-programmed like software, blurring the hardware/software distinction.

There are many solutions for realizing a beam former, which are listed below, but this report specifically regards realization of beam former using phased array antennas, so further on this report is focusing on that.

Beamforming solutions:

- Phased array antennas, which uses beamforming to steer the beam;
- Aperture synthesis;
- Inverse synthetic aperture radar (ISAR);
- Sonar, side-scan sonar;
- Synthetic aperture radar;
- Synthetic aperture sonar;
- Thinned array curse;
- Window function;
- Magneto encephalography (SAM);
- Microphone array;
- Zero-forcing precoding;
- Multi-beam echo sounder.

3. Beamforming Algorithm

3.1. Beamforming Based on Phased Array Antennas

In antenna theory, a phased array is an array of antennas in which the relative phases of the respective signals feeding the antennas are varied in such a way that the effective radiation pattern of the array is reinforced in a desired direction and suppressed in undesired directions.

An antenna array is a set of *M* spatially separated antennas. The number of antennas in an array can be as small as 2, or as large as several thousand. In general, the performance of an antenna array (for whatever application it is being used) increases with the number of antennas (elements) in the array; the drawback of course is the increased cost, size, and complexity.

The general form of an antenna array can be illustrated as shown in Figure 3.1. Given antenna array consists of 8 x 10 elements.

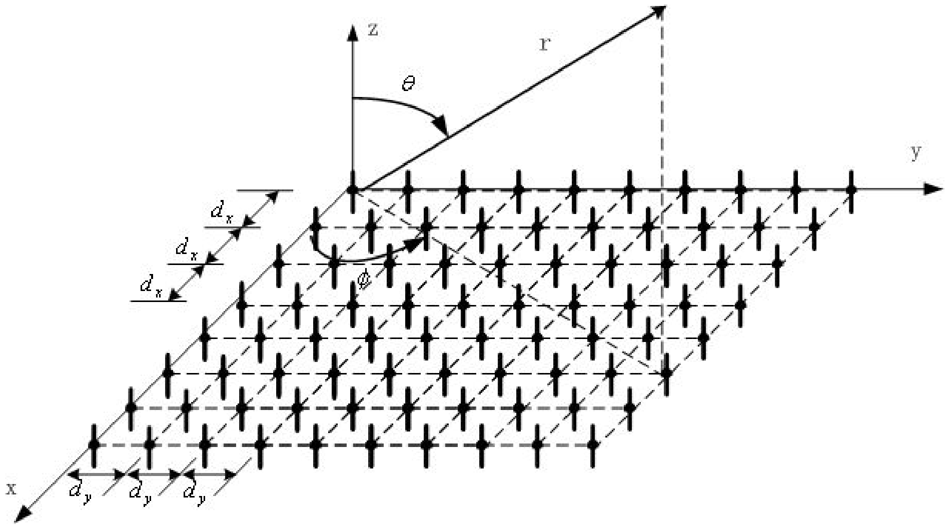

Fig. 3.1. General form of antenna array.

Let $X_1, X_2,...,X_M$ represent the output from antennas 1 thru *M*, respectively. The output from these antennas are most often multiplied by a set of *M* weights – $w_1, w_2,...,w_M$ - and added together as shown in Figure 3.2.

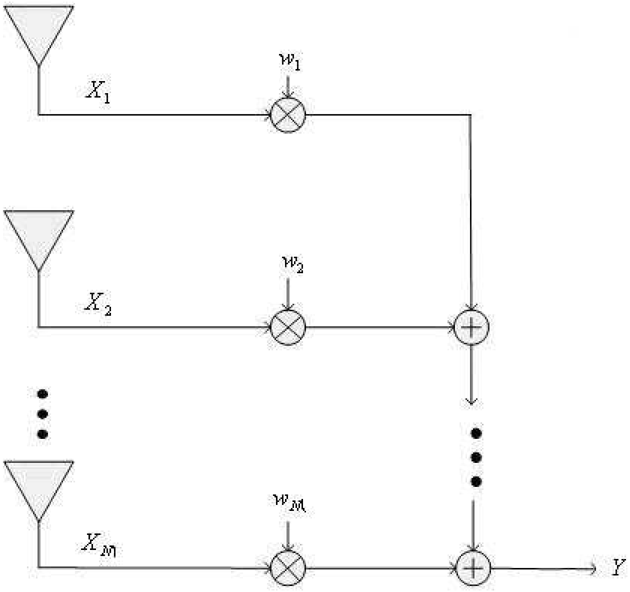

Fig. 3.2. Weighting and summing of signals from the antennas to form the output in a phased array.

According to the information mentioned before, the output of an antenna array can be written succinctly as:

$$Y = \sum_{m=0}^{M-1} w_m X_m$$

3.2. Realization of Beamforming Algorithm

In the first subsection a list of hardware for beamforming algorithm realization is presented. In the second subsection a more specific examples of beamforming algorithm realization are covered. First of all, to make it clearer a general example of beamforming algorithm is presented.

3.2.1. Implementation hardware

The realization of beamforming algorithm requires hardware with a very high computing power and a perfect solution for this necessity is field programmable gate array (FPGA). This is the main reason why FPGA based hardware is used in the implementation of antenna arrays. Still the most suitable FPGA series and model needs to be found regarding the specific application necessities.

The adoption of FPGA technology continues to increase, but it is still important to look inside the FPGA and appreciate how much is actually happening when block diagrams (algorithms) are compiled down to execute in silicon. Comparing and selecting hardware targets based on number of logic blocks, DSP slices, transceivers, multipliers type, block RAM size and the price is the best way to choose the right FPGA chip for the necessary application. Understanding resource usage is extremely helpful during development, especially when optimizing for size and speed. In the table given below a four types of FPGA series are listed to find the most suitable FPGA for beamforming algorithm realization.

Table 1

Overview of FPGA series resources

FPGA series		**Xilinx**		**Altera**	
		Kintex 7	**Virtex 7**	**Arria 10**	**Stratix 5**
Logic Blocks	Min	65 600	582 720	61 000	520 000
	Max	477 760	876 160	420 000	820 000
Multiplier type (Bits)		25 x 18	25 x 18	18 x 19 (floating-point)	18 x 18
Block RAM (Mb)	Min	4,7	28	9	51
	Max	34	51	53	52
DSP Slices	Min	240	1 260	156	512
	Max	1 920	2 520	1 700	3 900
Transceivers	Min	8	8	12	4
	Max	32	72	96	48
Price (Euro)	Min	116	50	170	191
	Max	4 820	35 000	**18 000**	21 000

3.2.2. General example

List of general variables:

num_of_ant => number of dipole antennas in a beam former;
observ_bw => observing bandwidth (MHz);
num_of_sig => number of antenna signals;
num_of_beams => number of formed beams;
num_of_complex => amount of complex numbers;
comp_prec => the precision of component (bits);
total_wf_bits => total number of bits provided for the weight factors (bits);
complex_mult_per_sec => total number of complex multiplications per second;
num_of_fpga => total number of FPGAs used in beam former;
mem_blocks => total number of FPGAs memory blocks = 720;
min_clk_freq => minimum clock frequency;
num_of_freq_ch => number of frequency channels across antenna signals;
num_of_complex_out_samp => number of complex output samples (bit/s);
bits_per_comp => the width of complex output sample per component (bits).

The fundamental mathematical operation describing a single-beam system (many-to-one) that it performs is

$$y(n) = \sum_{m=0}^{M-1} w_m \cdot x_m(n)$$
,

where $x_m(n)$ is the signal from the *m-th* antenna at sample time $n*T0$;
w_m is the weight factor for the *m-th* antenna;
$y(n)$ is the beam sample at time $n*T0$.

All quantities *y*, *x*, and *w* are complex numbers.

A system computing several beams $b = 0 \ldots B\text{-}1$ (many-to-many) can then be described as

$$y_b(n) = \sum_{m=0}^{M-1} w_{m,b} \cdot x_m(n)$$

Finally, if the antenna signals have been split into a number of frequency channels $c = 0 \ldots C\text{-}1$ the system can be described as

$$y_{b,c}(n) = \sum_{m=0}^{M-1} w_{m,b,c} \cdot x_{m,c}(n)$$

The beam former is polarization-agnostic, meaning, all antenna signals (num_of_sig) can be used for the computation of a given beam. Polarization can be included by setting the weights appropriately. The system must maintain a total of $M*B*C$ weight factors.

The basic steps for realization of beam former algorithm are calculations of:

1) The amount of complex numbers:

   num_of_complex = observ_bw * num_of_beams * num_of_sig

2) The total number of bits provided for the weight factors (bits):

   total_wf_bits = num_of_complex * comp_prec * 2

3) The total number of complex multiplications per second:

   complex_mult_per_sec = observ_bw * num_of_sig * num_of_beams

4) The minimum clock frequency:

   min_clk_freq = complex_mult_per_sec / (num_of_fpga * mem_blocks)

5) The number of frequency channels each FPGA needs to process across all antenna signals:

   num_of_freq_ch = observ_bw / num_of_fpga

6) Data rate into each beam former FPGA:

fpga_data_rate = num_of_freq_ch * num_of_sig*10^6 * num_of_complex_out_samp,

where **num_of_complex_out_samp** = num_of_sig / bits_per_comp.

The following two examples are calculated for specific parameters of FPGA series – *Arria 1*.

<u>Example 1</u>

The first example covered is an antenna array consisting of 36 (6 x 6) dipole antennas. This is a narrow-band beam former operating on frequency channels of around 1 MHz bandwidth.

The specifications are:

- 384MHz observing bandwidth;
- 36 dual-polarization receivers (72 antenna signals);
- 9 beams.

This beam former is polarization-agnostic, meaning, all 72 antenna signals can be used for the computation of a given beam. Polarization can be included by setting the weights appropriately. Calculations of beam former algorithm:

1) The amount of complex numbers:

384 MHz * 9 beams * 72 antenna signals = **248 832**

2) The total number of bits provided for the weight factors (bits), as the precision is set to 19 bits per component (according to FPGA specification):

248 832 * 19 * 2 = **9 455 616 bits**

3) The total number of complex multiplications per second:

$$384 * 10^6 * 72 * 9 = \mathbf{2.48832 * 10^{11}}$$

4) The minimum clock frequency:

As this example uses the slowest production version of the FPGA, which allows the complex multipliers to be used at up to 360 MHz. The complex multipliers are grouped into sub-arrays of 16 units, so that 7 complex multipliers remain unused per FPGA. In order to meet the computing requirements, a total of 48 FPGAs on 12 UniBoards are used. As beam former unit is divided into an array of independent beam former cores, beam former core uses 16 complex multipliers in parallel; accordingly, there are 47 beam former cores on the chip (47 * 16 = 752).

$$2.48832 * 10^{11} / (48 * 752) = \mathbf{6.89362\ MHz}$$

5) The number of frequency channels each FPGA needs to process across all antenna signals:

384 / 48 = **8** (frequency channels)

6) Data rate into each beam former FPGA:

We assume that channelization is done by polyphase filter banks, and that the complex output samples have a width of 16 bits per component.

$$8 * 72 * 10^6 * 4.5 = \mathbf{2.592 * 10^9}\ \textbf{bit/s}$$

$$512 / 16 = \mathbf{4.5}\ \textbf{bit/s}$$

<u>Example 2</u>

The second example is an antenna array consisting of 256 (16 x 16) dipole antennas. This is a narrow-band beam former operating on frequency channels of around 1 MHz bandwidth.

The specifications are:

- 384MHz observing bandwidth;
- 256 dual-polarization receivers (512 antenna signals);
- 64 beams.

The beam former is polarization-agnostic; accordingly all 512 antenna signals can be used for the computation of a given beam. Calculations of beam former algorithm:

1) The amount of complex numbers:

384 MHz * 64 beams * 512 antenna signals = **12 582 912**

2) The total number of bits provided for the weight factors (bits), as the precision is set to 19 bits per component:

12 582 912 * 19 * 2 = **478 150 656 bits**

3) The total number of complex multiplications per second:

$$384 * 10^6 * 512 * 64 = \mathbf{1.2582912 * 10^{13}}$$

4) The minimum clock frequency:

As this example uses the slowest production version of the FPGA, which allows the complex multipliers to be used at up to 360 MHz. The complex multipliers are grouped into sub-arrays of 16 units, so that 7 complex multipliers remain unused per FPGA. In order to meet the computing requirements, a total of 48 FPGAs on 12 UniBoards are used. As beam former unit is divided into an array of independent beam former cores, beam former core uses 16 complex multipliers in parallel; accordingly, there are 47 beam former cores on the chip (47 * 16 = 752).

$$1.2582912 * 10^{13} / (48 * 752) = \mathbf{348.6\ MHz}$$

5) The number of frequency channels each FPGA needs to process across all antenna signals:

384 / 48 = **8** (frequency channels)

6) Data rate into each beam former FPGA:

We assume that channelization is done by polyphase filter banks, and that the complex output samples have a width of 16 bits per component.

$$8 \quad * 512 * 10^6 * 32 = \mathbf{1.31072 * 10^{11}}\ \textbf{bit/s,}$$

$$512 / 16 = \mathbf{32}\ \textbf{bit/s}$$

From doing both examples with the same number of FPGAs, UniBorads and series, we assume conservatively that for the first case (example 1) a net data rate of 0.1 Gb/s can be achieved per backplane lane, but for the second case a net data rate of 10 Gb/s. Thus, for the second case each beam former FPGA needs at least 14 such interfaces. Each beam former-FPGA provides a total of 72 such backplane interfaces, so this is not a limitation.

Comparing calculated minimum clock frequencies in both cases, in the first case less than a half of the total amount of FPGAs (48) could be used to use this beam former effectively.

The results in both examples have a great difference in the total number of bits provided for the weight factors: in example of case with 36 antennas - 9 455 616 bits, but in example 2 of case with 256 antennas - 478 150 656 bits; as the weighting is one of the main factors for antenna array adaption, this shows that the size (number of components) of antenna array matters to improve its performance.

Analysing the information given in table 1, it can be concluded that the best and most appropriate FPGA series for antenna array implementation would be Altera – *Arria 10*. *Arria 10* is the best choice from all of the listed FPGAs to have effective use regarding its resources and price.

4. Conclusions

1. Radio interferometry gives engineers the possibility of building bigger radio telescopes, instead of using a large single dish antenna; it combines the signals received from many different antennas to form a virtual radio telescope that is the combination of all the small antennas.
2. The LOw Frequency ARray (LOFAR) radio telescope is an example of this new generation of telescopes.
3. Beamforming permits to give directionality to an array of non-directional antennas.
4. The beamforming technique used in receiving involves combining delayed signals from each antenna at slightly different, so that every signal reaches the output at exactly the same time, making one loud signal, as if the signal came from a single, very sensitive receiver.
5. Actual number of backplane interfaces depends on the architecture of the polyphase filter banks, and on how antenna signals are connected to the filter bank FPGAs.
6. Particular considerations can be used to derive architecture and size of the beam former; because of the multitude of hardware constraints it is very hard to give general rules for scaling the system.

5. References

1. *Alessio Sclocco.* „Radio Astronomy Beamforming on GPUs", 2011;
2. *W. C. Barott, O. Milgrome, M. Wright, D. MacMahon, T. Kilsdonk*: „Real-Time Beamforming Using High-Speed FPGAs at the Allen Telescope Array", 2011;
3. *Guenter Knittel.* „Uniboard^2 Beam Former Firmware Design Document", 2014;
4. *Barry D. Van Veen and Kevin M. Buckley*. "Beamforming: A Versatile Approach to Spatial Filtering", 1988;
5. *Lei Wang.* "Array Signal Processing Algorithms for Beamforming and Direction Finding", 2009;
6. An Introduction to Beamforming.
   URL: http://staff.washington.edu/aganse/beamforming/beamforming.html;
7. Antenna Arrays.
   URL: http://www.antenna-theory.com/arrays/main.php;
8. Phased Array.
   URL: http://en.wikipedia.org/wiki/Phased_array;
9. Beamforming.
   URL: http://en.wikipedia.org/wiki/Beamforming.