Realizing Sample Matrix Inversion (SMI) in Digital BeamForming (DBF) System

Hao Lei, Zaiping Nie and Feng Yang
University of Electronic Science and Technology of China
NO.4, Section2, North Jianshe Road
Chengdu, Sichuan 610054 China

***Abstract*-Digital beamforming is a kind signal processing in adaptive array antennas, based on the disposal of data produced by sampling signal. DBF can adjust beam pattern based on the arrival direction of signal, and produce pattern nulling in the direction of disturb signal. Sample matrix inversion(SMI). Algorithm is one kind method in DBF. This paper will introduce my work on simulating and verifying QRD-SMI algorithm based on Virtex 5, an popular FPGA chip produced by XILINX company. QR matrix decomposition is used to transfer matrix to a upper triangle matrix which is more easily to solve the inverse matrix.**

Keywords: Digital beamforming, QR-SMI, matrix inversion, FPGA.

I. INTRODUCTION

SMI Algorithm is based on the standard that make signal to interference-plus-noise ratio(SINR)[1] maximum, which the desired array signal and the unwanted signal have largest proportion. Linearly constrained minimum variance (LCMV) optimal weight vector satisfies the following linear equation

$$R_{XX}\mathrm{w}_{opt} = \mu s^{*} \tag{1}$$

Where R_{xx} is the sampling covariance matrix, μ is any proportionality constant, s is operation vector of required signal, that is static weight which control the beam in the target direction[2]. Based on the maximum likelihood criterion, N signal sampled data may constitute the best estimate R_{xx}. In practical engineering, inversing R_{xx} will consume a lot of time and hardware resources, the method to reduce the computational and hardware resources is using the Givens rotation to realize matrix QR decomposition. Changing the problem that solving weight vector W into a problem solving triangular linear equation. Since the covariance matrix R_{xx} can be expressed as $R_{XX} = X_n^H X_n$, Xn is a $n \times M$ signal sampled matrix which n and M present the number of sample data and the number of array antennas respectively. We can get the equation $X_n^H X_n w = S$. If there is a unitary matrix Q will triangulate matrix X_n,

$$QX_n = \begin{bmatrix} A_n \\ 0 \end{bmatrix} \tag{2}$$

where An is M×M upper triangular matrix, an important equation can be deduced as follow:

$$X_n^H X_n = X_n^H Q^H Q X_n = (QX_n)^H QX_n = [A_n^H, 0^H]\begin{bmatrix} A_n \\ 0 \end{bmatrix} = A_n^H A_n \tag{3}$$

The advantage of these processes is that we have lots of methods to get weight vector more easily. Technical difficulty is how to change the sampled signal matrix X_n into a triangular matrix A_n in FPGA. Givens rotation is an appropriate method. Givens rotation is particularly suitable for adaptive array applications, because it better to be applied in FPGA.

II. GIVENS ROTATION

Complex Givens rotation can be presented as the following elementary transformations:

$$\begin{bmatrix} C & S^{*} \\ -S & C \end{bmatrix}\begin{bmatrix} 0 & .. & \mathrm{x_i} & \mathrm{x_{i+1}} & ... & \mathrm{x_k} \\ 0 & .. & \mathrm{y_i} & \mathrm{y_{i+1}} & ... & \mathrm{y_k} \end{bmatrix} = \begin{bmatrix} 0 & .. & \mathrm{x_i'} & \mathrm{x_{i+1}'} & ... & \mathrm{x_k'} \\ 0 & . & 0 & \mathrm{y_{i+1}'} & ... & \mathrm{y_k'} \end{bmatrix} \tag{4}$$

Where $G = \begin{bmatrix} C & S^{*} \\ -S & C \end{bmatrix}$ mustbe a unitary matrix simultaneously, so matrix G must meet following equation

$$\begin{cases} -Sx_i + Cy_i = 0 \\ S^{*}S + C^{*}C = 1 \\ C^{*} = C \end{cases} \tag{5}$$

Assuming X (n-1) have already achieved triangulation. When $\mathrm{x^T}=[\mathrm{x_1(n)},...,\mathrm{x_M(n)}]$ is being inputed, $n \times M$ sampled data matrix can be written as follow:

$$X(n) = \begin{bmatrix} X(n-1) \\ x^{\mathrm{T}}(n) \end{bmatrix} \tag{6}$$

Rotating matrix X(n) by a new n×n unitary matrix

$$\overline{Q}(n-1) = \begin{bmatrix} Q(n-1) & 0_{\mathrm{n\text{-}1}} \\ 0_{n-1}^{T} & 1 \end{bmatrix} \tag{7}$$

An new equation can be presented as follows:

$$\overline{Q}(n-1)X(n) = \begin{bmatrix} Q(n-1)X(n-1) \\ x^{T}(n) \end{bmatrix} = \begin{bmatrix} R(n-1) \\ 0 \\ x^{T}(n) \end{bmatrix} = \begin{bmatrix} z_{11} & z_{12} & \cdots & z_{1M} \\ & z_{22} & \cdots & z_{2M} \\ & & \cdots & \cdots \\ & & & z_{MM} \\ 0 & \cdots & \cdots & \cdots \\ x_1(n) & x_2(n) & \cdots & x_M(n) \end{bmatrix} \tag{8}$$

Defining the first givens rotation matrix G1(n) that is eliminating the *x1(n)*, so we get equation (9)

$$G_1(n)\overline{Q}(n-1)X(n)=\begin{bmatrix} z_{11}' & z_{12}' & \cdots & z_{1M}' \\ & z_{22} & \cdots & z_{2M} \\ & & \cdots & \cdots \\ & & & z_{MM} \\ 0 & \cdots & \cdots & \cdots \\ 0 & x_2^{(1)}(n) & \cdots & x_M^{(1)}(n) \end{bmatrix} \quad (9)$$

Defining the second givens rotation[3] matrix *G2(n)* , *G2(n)* is used to eliminate $x_2^{(1)}(n)$, a new equation can be written as follow:

$$G_2(n)G_1(n)\overline{Q}(n-1)X(n)=\begin{bmatrix} z_{11}' & z_{12}' & \cdots & \cdots & z_{1M}' \\ & z_{22}' & \cdots & \cdots & z_{2M}' \\ & & & \cdots & \\ & & & & z_{MM} \\ 0 & & & & \\ 0 & 0 & x_3^{(1)}(n) & \cdots & x_M^{(2)}(n) \end{bmatrix} \quad (10)$$

After M times rotations, a new upper triangular matrix[4] can be presented as follow:

$$G_M(n)\ldots G_2(n)G_1(n)\overline{Q}(n-1)X(n)=G(n)\begin{bmatrix} R(n-1) \\ 0 \\ x^T(n) \end{bmatrix}=\begin{bmatrix} R(n) \\ 0 \end{bmatrix} \quad (11)$$

The above describes a recursive method to get the upper triangular matrix. But these are based on the formula derivation, how to achieve Givens rotation in circuit or in FPGA chip？ Achieving givens rotation in FPGA is the focus of this paper. As we know, it is difficult to do some matrix processing in circuit or in FPGA. FPGA is based on parallel thought, FPGA has no integrated math library to help us to do some math processes. Compared to floating-point arithmetic, fixed-point arithmetic is more complex. Fixed-point arithmetic needs to be considered data overflow, fractional process. Next I will introduce Systolic array to implement givens rotation in FPGA.

III. SYSTOLIC ARRAY AND CORDIC

Systolic array[5] is a parallel pipelined and high-speed signal processing algorithm. Systolic array have lots of advantages, including modular and locality. Systolic array is based on multiprocessor architecture.

All processors have rhythm synchronization. Systolic achieve a high degree of parallelism and pipelining and local communication between processors. Now I will discuss how to achieve givens rotation by systolic array.

Realizing upper triangulation by systolic including two important modules, boundary element and internal element. A 4x4 systolic array is presented in Fig1

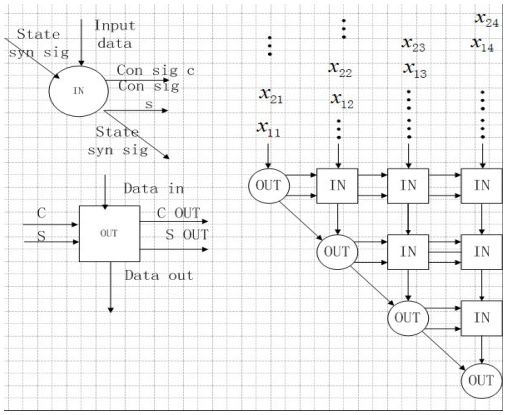


Fig. 1. 4x4 systolic arrays and its boundary and internal element

Boundary element is core unit of systolic array, located on the diagonal of systolic array. Boundary element have two main core functions, firstly it must finishing processing the input data to zero, secondly provide control signals(Con sig c, Con sig s) to internal element produced by zero processing. Internal element is located in the upper right diagonal of systolic array, its function is receiving control signal generated from the boundary element and controlled by Con sig c and Con sig s. 4x4 systolic arrays can process the input data sampled from 4 antennas, but we need control the order data enters shown in Fig1.

How to design boundary element and internal element. They must finish function and cost minimal resources. I introduce CORDIC algorithm to achieve this goal. Complex operation such as multiplication and division, can be achieved by shifting processing and addition and subtraction merely. This greatly reduces the implementation complexity required by lots of processing units of the systolic array.

The full name of CORDIC is Coordinate Rotation Digital Computer, be invented by Jack. E. Volder in 1959.Now CORDIC has become a modern numerical model in hardware acceleration. Now introducing the basic principles on cordic[6].

Let the vector (x_0, y_0) clockwise rotation θ , the new coordinates(x_1, y_1) can be written as follows:

$$\begin{cases} x = x_0\cos\theta + y_0\sin\theta \\ y = -x_0\sin\theta + y_0\cos\theta \end{cases} \quad (12)$$

Where above formula can be changed to the follow equation

$$\begin{cases} x = K(x_0 + y_0\tan\theta) \\ y = K(-x_0\tan\theta + y_0) \\ z_1 = z_0 - \theta \\ K = (1+\tan^2\theta)^{-1/2} \end{cases} \quad (13)$$

The original rotation can be instead of rotating a series of multiple rotations

$$\theta = \delta_1 a_1 + \delta_2 a_2 + \ldots + \delta_n a_n \qquad \delta_i = \pm 1 \quad (14)$$

δ_i presents the direction of the base angle rotation, if $\tan a_i = 2^{-i}$, multiplying 2^{-i} can be get by shifting right i bit, this operation is extremely simple for digital circuit, so a

important equation can be deduced according to (13):

$$\begin{cases} x_{i+1} = (x_i + y_i \delta_i 2^{-i}) \\ y_{i+1} = (-x_i \delta_i 2^{-i} + y_i) \\ z_{i+1} = z_i - \delta_i \tan^{-1}(2^{-i}) \\ K_i = \prod_{i=0}^{n} (1 + 2^{-2i})^{-1/2} \end{cases} \quad (15)$$

When n is large enough, equation15 can make z_i to be close to zero, so the angle can be written as follow:

$$\theta = \sum_{i=1}^{n} d_i \tan^{-1}(2^{-i}) = -z \quad (16)$$

When we rotate this vector to x-axis, the value of x-axis is the amplitude of original vector. Simultaneously Z presents the angel the original vector rotated to the x-axis. Now the recursive process is not a pure rotation, finally we need to make compensate to the amplitude of vector. When n is large enough, K is close to 0.6072, a constant[7]. When all rotation finished, we can process compensation by making shifting and addition and subtraction. Because the sampled data is complex number, we must rotate the complex number to a real number firstly. So the Givens rotation of a complex number can be presented by equation (17)

$$G(n,i) = \begin{pmatrix} c_i & s_i^* \\ -s_i & c_i \end{pmatrix} = \begin{pmatrix} 1 & \\ & \exp(j\varphi_i) \end{pmatrix} \begin{pmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i & \cos\theta_i \end{pmatrix} \begin{pmatrix} 1 & \\ & \exp(-j\varphi_i) \end{pmatrix}$$

$$\varphi_i = \tan^{-1} \frac{\mathrm{Im}(x_i)}{\mathrm{Re}(x_i)}, \theta_i = \tan^{-1} \frac{|x_i|}{r_i} \quad (17)$$

IV. Simulation In FPGA

FPGA has strong parallel and pipeline properties which are extremely ideal for systolic array. Simultaneously, systolic requires globe clock which FPGA can provide. The basic structure of QRD-SMI is presented in Fig2.

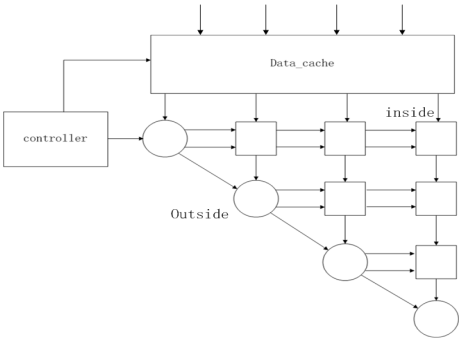


Fig. 2. Basic structure of 4x4 QRD-SMI

The implement of QRD-SMI including four basic module, control module, data buffer module, outside module and inside module. The program is described by VERILOG which is specialized hardware description language. The block diagram of outside module is presented in Fig.3.

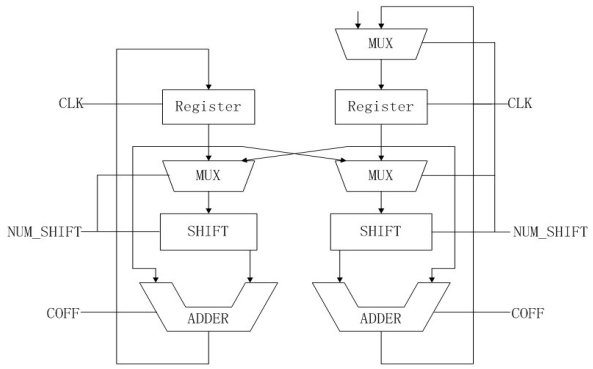
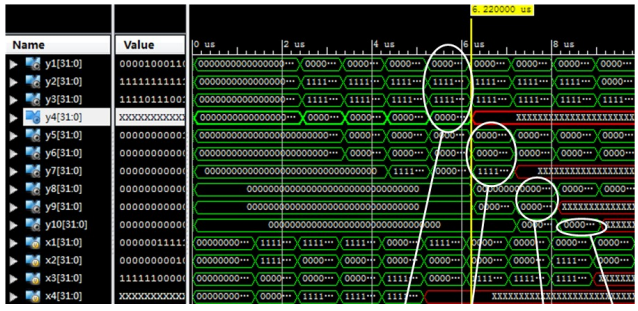


Fig. 3. The block diagram of outside module.

Next I will describe the operation of outside module. When NUM_SHIFT=0, the input data is registered in input register. In the next 24 system clock, the module does not accept data. Before 20 cycles, multiplexer and shift register design shift bits, simultaneously adder or subtraction is selected by the COFF signal. In the last 4 cycles, module must finish amplitude correction. In the 24th cycle, NUM_SHIFT will be zero and module will receive external data again.

In the outside module, the COFF signal is designed by iteration results. If the data is greater than zero, COFF is high, the vector will clockwise rotate. On the contrary, COFF is low, the vector will anticlockwise. At the same time, the signal of NUM-SHIFT and COFF will be conveyed to inside module. These signals will control shift operation and add operation in inside module. Multiple processors will work together through this mode. For complex rotation, the module must rotate the imaginary part to zero. It takes 23 cycles again. So the operation for complex data takes 47 cycles in total.

Testing this program in ISE13.1 platform. In Fig.4, x1, x2, x3, x4 constitute 4x4 input matrix, the high 16 bits are real part of signal, the low 16 bits are imaginary part. y1—y10 are outputs of upper triangular matrix. We can compare matrix RR(result of our algorithm) with matrix R (result of QR function in MATLAB), we can find deviation is in acceptable range.



$$RR = \begin{bmatrix} 200 & -3+200i & -201+1i & 200i \\ 0 & 4 & 1+5i & 1i \\ 0 & 0 & 1 & 1+2i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$R = \begin{bmatrix} 200.62 & -3.23+200.96i & -201.31+0.71i & 0.18+200.12i \\ 0 & 3.84 & 1.23+5.14i & -0.41+0.77i \\ 0 & 0 & 1.40 & 1+2.02i \\ 0 & 0 & 0 & 0.70 \end{bmatrix}$$

Fig. 4. The simulated results in ISE platform.

Assuming a simulation model, signal frequency is 2GHz, the direction of desired signal is 0°, SNR is 0dB. The direction of disturb signal is -20°, SNR is 40dB , the number of input data are 64. ISE simulation data is exported to MATLAB to produce image of pattern as Fig.5, we can get a similar result.

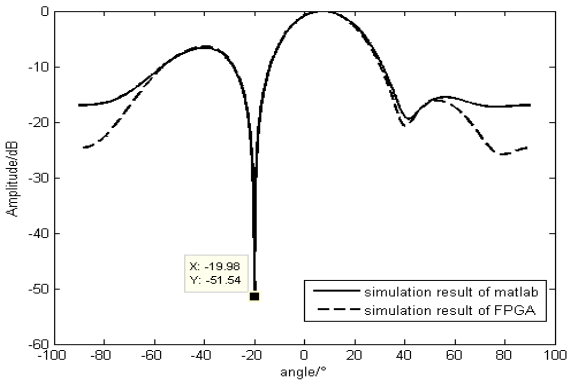
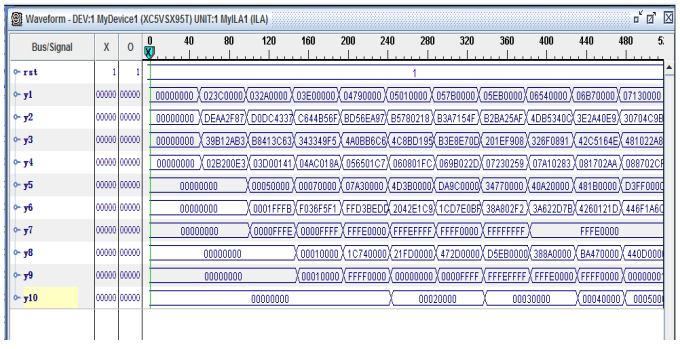


Fig.5. Comparing givens rotation result with matlab result

Downloading the program into Virtex5 chip, observing the signal in chip by internal logic analyzer called CHIPSCOPE. Simulating input signal through VIO, ILA, ICON IPCORE. Observing the output in simulation and the output in Virtex5, We can verify the implementation of the program downloaded in the chip. Two results are shown in Fig6. By comparison, two results are identical. This will verify the correctness of the program. But the input data is limit, it is difficult for us to verify the large amount of input data. The frequency of system clock is 50MHz.



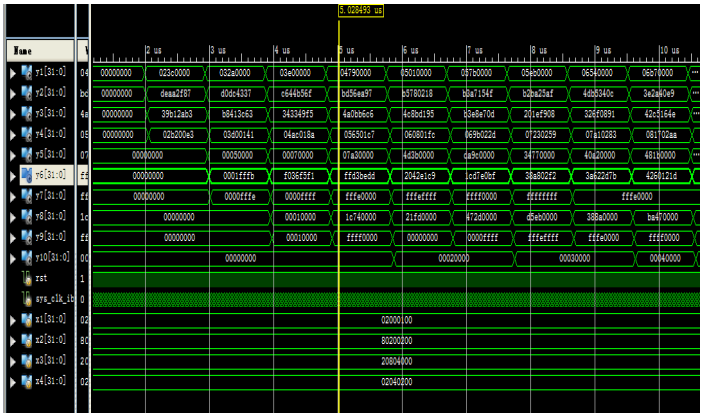


Fig.6. Output in chip and the output of simulation status

V. CONCLUSION

The process which QRD-SMI is realized in FPGA is introduced. We use the CORDIC algorithm to complete the Givens rotation. However, one operation cost 47 cycles, system's ability to handle high data rate is limited. The program need to optimized to meet faster system clock. If the program can run in 100MHz, two times the data rate can be handle. In addition, reducing cycles of one process by optimizing the frame of program.

REFERENCES

[1] Yu Lei, Wei Liu, Langley R., "SINR Analysis of the Subtraction-based SMI Beamform," *IEEE Trans. on Signal Processing* vol. 58, pp. 5926-5932. 2010.

[2] Wang, A. K, and Leary, J, "SMI based beamforming algorithms for TDMA signals," *the Thirty-First Asilomar Conference on*, vol. 2, pp.1326-1330. 1997.

[3] Min-Woo Lee and Ji-Hwan Yoon, "High-speed tournament givens rotation-based QR Decompositon Architecture for MIMO Reciever," *IEEE International Symposium on.*,, pp.21-24. Feb. 2012.

[4] Bienati, N. Spagnolini, U. Zecca, M "An adaptive blind signal separation based on the joint optimization of Givens rotations," acoustics, Speech and signal processing.2001 *IEEE International Conference on*, vol.5, pp.2809-2812. Aug. 2001.

[5] Aslan, S, Niu, S., Saniie, J., "FPGA implementation of fast QR decomposition based on givens rotations., vol.6, pp.470-473. 2012.

[6] Chih-Hsiu Lin.An-Yeu Wu, "Mixed-scaling-rotation CORDIC algorithm and architecture for high performance vector rotational DSP application," *Circuits and system I: regular Papers, IEEE transaction on.*, vol.55, pp. 2385-2396. Apr. 2005.

[7] Cheng-Shing Wu, An-Yeu,, "Modified vector rotational cordic algorithm and architecture,"*Circuits and systems : Analog and digital signal processing, IEEE transaction on.*, vol.48, pp.546-561. Dec. 2001.