# Complex Matrix Factorizations with CORDIC Arithmetic

Joseph R. Cavallaro*
Anne C. Elster**

TR 89-1071
December 1989

Department of Computer Science
Cornell University
Ithaca, NY 14853-7501

*Dept. of Electrical and Computer Engineering, Rice University, Houston, TX 77251.
Supported in part by the National Science Foundation under Research Initiation Award MIP-8909498.
**Dept. of Computer Science, Upson Hall, Cornell University, Ithaca, NY 14853.
Supported in part by the U.S. Army Research Office through the Mathematical Sciences Institute, Cornell University.

# Complex Matrix Factorizations with CORDIC Arithmetic*

Joseph R. Cavallaro[†] Anne C. Elster[‡]

December 1989

**Abstract**

Matrix factorizations are important in many real-time signal processing applications. In order to improve the performance of these algorithms, special-purpose VLSI processor arrays are being developed. Recently, the Coordinate Rotation Digital Computer (CORDIC) algorithms have been applied to the QR Decomposition (QRD) and the Singular Value Decomposition (SVD). In this paper, the CORDIC arithmetic algorithms are extended to deal with complex data. Novel CORDIC VLSI architectures for the QRD of a complex matrix, the Eigenvalue Decomposition of a Hermitian matrix, and the SVD of a complex matrix are presented. These architectures are suitable for VLSI implementation.

**Index Terms:**
Complex Matrix Factorizations, CORDIC Algorithms, QR Decomposition, Real-Time Digital Signal Processing, Singular Value Decomposition, Special-Purpose VLSI Processor Architectures, Systolic Arrays.

*Submitted to the *IEEE Transactions on Computers*. The complex CORDIC QR approach was originally presented by the authors at the Conference on Approximation Theory and Numerical Linear Algebra, Kent, Ohio, April 1989.

[†]Dept. of Electrical and Computer Engineering, Rice University, Houston, Texas 77251. Supported in part by the National Science Foundation under Research Initiation Award MIP-8909498.

[‡]Dept. of Computer Science, Upson Hall, Cornell University, Ithaca, New York 14853. Supported in part by the U.S. Army Research Office through the Mathematical Sciences Institute, Cornell University.

# 1 Introduction

Real-time signal processing and image processing algorithms use various matrix factorizations steps. In order to improve the time performance of these specialized algorithms, custom VLSI processor arrays are being designed. These arrays use parallel numerical algorithms and incorporate high-speed arithmetic. Many problems in signal processing utilize complex matrix factorizations and can benefit from a complex CORDIC SVD array. In particular, several adaptive beamforming algorithms [13,14] which determine the direction or bearing of a signal source, require complex matrix factorizations.

The coordinate rotation digital computer algorithms (CORDIC) are useful for elementary function evaluation. In particular, CORDIC is capable of vector rotation by an angle, $\theta$, and inverse tangent and radius calculation. The algorithms are based upon a regular scheme that only uses shifts and additions. These features of CORDIC are very beneficial in designing custom hardware for parallel matrix operations.

The most direct application of CORDIC is to the Givens rotation. The Givens rotation is an orthogonal and norm preserving rotation from which we can build systolic arrays to perform several matrix decompositions. The QR decomposition involves a sequence of one-sided rotations, whereas the eigenvalue decomposition (limited to symmetric matrices) is a one angle, two-sided rotation approach. This decomposition uses the Jacobi algorithm which requires a square array and the ability to exchange matrix data as several sweeps are performed. A single rotation angle for each $2 \times 2$ submatrix is required. The real Singular Value Decomposition (SVD) also utilizes the Jacobi algorithm. Here two rotation angles are computed for each diagonal $2 \times 2$ submatrix.

An architecture for the real QRD which uses CORDIC arithmetic was presented by Ahmed, Delosme, and Morf [1]. Recently, CORDIC was applied to the real SVD by Cavallaro and Luk [3]. Both of these architectures use a group of interconnected CORDIC modules to first calculate rotation angles, and then apply them via vector rotations. This eliminates the need for multiplication and square root units.

In this paper, these CORDIC algorithms will be extended to the complex case. We will discuss how the Eigenvalue Decomposition of a symmetric matrix becomes the Eigenvalue Decomposition of a Hermitian matrix, and how the QRD and the SVD take on complex form. The goal is to formulate the algorithms in terms of inverse tangent and vector rotation operations by using only unitary rotation matrices.

The rotations in the CORDIC arithmetic algorithms extend elegantly to the complex case. In section 2, the CORDIC algorithms are briefly reviewed. Section 3 introduces the Complex Givens rotation as an extension of the real Givens rotation. In section 4, CORDIC is applied to the complex Givens rotation. The remaining sections discuss in turn the application of CORDIC to the complex QRD, Eigenvalue Decomposition of a Hermitian Matrix, and the complex SVD. Architectures are presented and an evaluation of the area and time complexity is also included.

## 2 CORDIC Algorithms

The CORDIC algorithms provide a fast hardware implementation of the inverse tangent and vector rotation operations. These algorithms have traditionally been described for the real case for a rotation in a two-dimensional plane. Volder [15] and Walther [16] provided the initial description of CORDIC. The CORDIC algorithms are based upon defining a vector $(x_0, y_0)$ in the 2-plane, and then applying a rotational transformation. That is, the vector $(x_0, y_0)$ is rotated through an angle $\theta$, in the clockwise direction, to $(x_0', y_0')$. In general, the CORDIC equations can describe a rotation in one of three modes: circular, linear, or hyperbolic. For the QRD, Eigenvalue Decomposition, and the SVD, the rotations are in the circular mode and the real rotation equations are:

$$\begin{bmatrix} x_0' \\ y_0' \end{bmatrix} = R(\theta) \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} . \tag{1}$$

In the CORDIC algorithms the rotation angle is decomposed into a sequence of $n$ known smaller angles, such that

$$\theta = \pm\,\theta_0 \pm \theta_1 \ldots \pm \theta_{n-1} = \sum_{i=0}^{n-1} \delta_i \theta_i \ , \tag{2}$$

where $\theta_i > 0$ and $\delta_i = \pm 1$. From the geometry of the rotations, it is clear that the result of $n$ rotations using the sequence of $\theta_i$'s is equivalent to that of one rotation using $\theta$. The number of known angles in the sequence determines the accuracy of the CORDIC algorithms. In order to achieve $n$ bits of accuracy, at least $n$ rotations must be performed.

In order to convert the rotation equations into a form which is suitable for VLSI implementation, the equations are first divided by the cosine. A further simplification is then made by setting $\tan\theta_i = \beta^{-i}$ where $\beta$ is the machine radix. In most applications, binary arithmetic is used, so $\beta = 2$, and therefore multiplication by $\tan\theta_i$ becomes a simple arithmetic shift operation. With this modification, the recurrence equations become:

$$\frac{x_{i+1}}{\cos\theta_i} = x_i + y_i 2^{-i} \tag{3}$$

$$\frac{y_{i+1}}{\cos\theta_i} = y_i - x_i 2^{-i} .$$

For example, when $i = 0$, then $2^{-i} = 1$ and $\theta_i = \tan^{-1}(2^{-i}) = 45°$. Again, for $i = 1$, $\theta_i = 26.7°$. It can be seen that as $i$ increases, $\theta_i$ decreases toward 0.

In practice, the cosine terms are collected into a scale factor constant which is removed in a final correction step. The recurrence equations then become:

$$x_{i+1} = x_i + \delta_i y_i 2^{-i} \tag{4}$$

$$y_{i+1} = y_i - \delta_i x_i 2^{-i}$$

$$z_{i+1} = z_i + \delta_i \theta_i \,.$$

It should be noted that $\delta_i$ determines the direction of rotation at a given step and that the $z$ equation accumulates the total rotation, $\theta$. These equations can be implemented efficiently in VLSI using registers, barrel shifters, adders, and a small ROM. The CORDIC algorithms can be further pipelined through the use of on-line arithmetic [9].

# 3 Complex Givens Rotations

The Givens rotation [12] is an important technique which is used to selectively introduce a zero into a matrix. Givens rotations can be used in the QR Decomposition and are similar to the rotations used in the Jacobi method for Eigenvalue Decomposition and the SVD. A real Givens rotation is given by:

$$\begin{bmatrix} c & s \\ -s & c \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} r \\ 0 \end{bmatrix} \quad \text{where } \theta = \tan^{-1}\left(\frac{b}{a}\right) \,. \tag{5}$$

Rotations can be generalized by considering the case of complex arithmetic. A complex Givens rotation can be described by two rotation angles as formulated in Coleman and Van Loan [6],

$$\begin{bmatrix} c_1 & s_1(c_2 + i\, s_2) \\ -s_1(c_2 - i\, s_2) & c_1 \end{bmatrix} \begin{bmatrix} a_1 + i\, a_2 \\ b_1 + i\, b_2 \end{bmatrix} = \begin{bmatrix} r_1 + i\, r_2 \\ 0 \end{bmatrix} . \tag{6}$$

The above rotation matrix is derived from applying the simple unitary transformation:

$$U = \begin{bmatrix} e^{i\theta_a} & 0 \\ 0 & e^{i\theta_b} \end{bmatrix} \tag{7}$$

to get $r$, and a real Givens rotation(5) to zero out the second component. Notice that the result is now real. However, in order to avoid four complex rotations, the complex conjugate of (7) is applied to the left side of the real Givens rotation, giving the complex Givens rotation in (6).

The angles $\theta_1$ and $\theta_2$ can be determined from the input vector as follows:

$$R_a = \sqrt{a_1^2 + a_2^2} \,, \qquad \theta_a = \tan^{-1}\left(\frac{a_2}{a_1}\right) \tag{8}$$

$$R_b = \sqrt{b_1^2 + b_2^2} \,, \qquad \theta_b = \tan^{-1}\left(\frac{b_2}{b_1}\right)$$

Then from the above angles and radii,

$$\theta_1 = \tan^{-1}\left(\frac{R_b}{R_a}\right) \,, \qquad \theta_2 = \theta_a - \theta_b \,. \tag{9}$$

# 4 Application of CORDIC to Complex Givens Rotations

The CORDIC algorithms describe a vector, $(a, b)$, in terms of the radius, $R = \sqrt{a^2 + b^2}$, and angle, $\theta = \tan^{-1}(b/a)$. It will be important to exploit this ability in the Complex CORDIC algorithms. The real Givens rotation can be rather simply computed by CORDIC. The angle $\theta$ is found using the inverse tangent mode. The rotation can then be applied by using the vector rotation mode.

Stating Euler's Formula as:

$$e^{i\theta} = \cos\theta + i \sin\theta , \tag{10}$$

a complex number, $z = z_1 + i\, z_2$ can then be written as:

$$z = R_z e^{i\,\theta_z} \quad \text{where} \quad R_z = \sqrt{{z_1}^2 + {z_2}^2} , \quad \theta_z = \tan^{-1}\left(\frac{z_2}{z_1}\right) . \tag{11}$$

These facts allow for the use of the CORDIC inverse tangent and vector rotation algorithms to calculate and apply complex Givens rotations without the need for multiplication and square root.

The above algorithm for finding $\theta_1$ and $\theta_2$ can be converted into the interconnected CORDIC modules shown in Figure 1. The input vector is $(a_1 + i\,a_2, b_1 + i\,b_2)$. At the first time step, two CORDIC modules in parallel find the angle and radius expressions for each of the complex numbers. At the second time step, a CORDIC module finds $\theta_1$ from the computed radii, and $\theta_2$ by subtracting $\theta_b$ from $\theta_a$.

The two rotation angles can then be applied using only CORDIC modules. Figure 2 shows the CORDIC implementation. In the first time step, a rotation by $\theta_2$ is applied to both complex numbers in parallel. These results are used in the second time step, along with $\theta_1$ to produce the final output vector, $(x_1 + i\,x_2), (y_1 + iy_2)$.

# 5 Complex CORDIC QR Decomposition Array

The QR Decomposition algorithm can be formulated in terms of Givens rotations. The algorithm applies orthogonal rotations to make a matrix upper triangular. The QRD systolic array has been presented by Gentleman and Kung [11]. The CORDIC algorithms have been applied to the QRD by Ahmed [1]. Figure 3 shows a triangular array for the QR decomposition. Each processor contains a CORDIC unit which calculates and applies the appropriate rotation.

The basic real Givens rotation can be replaced by the complex Givens rotation shown above. The complex CORDIC implementation can be used here for each QR processor

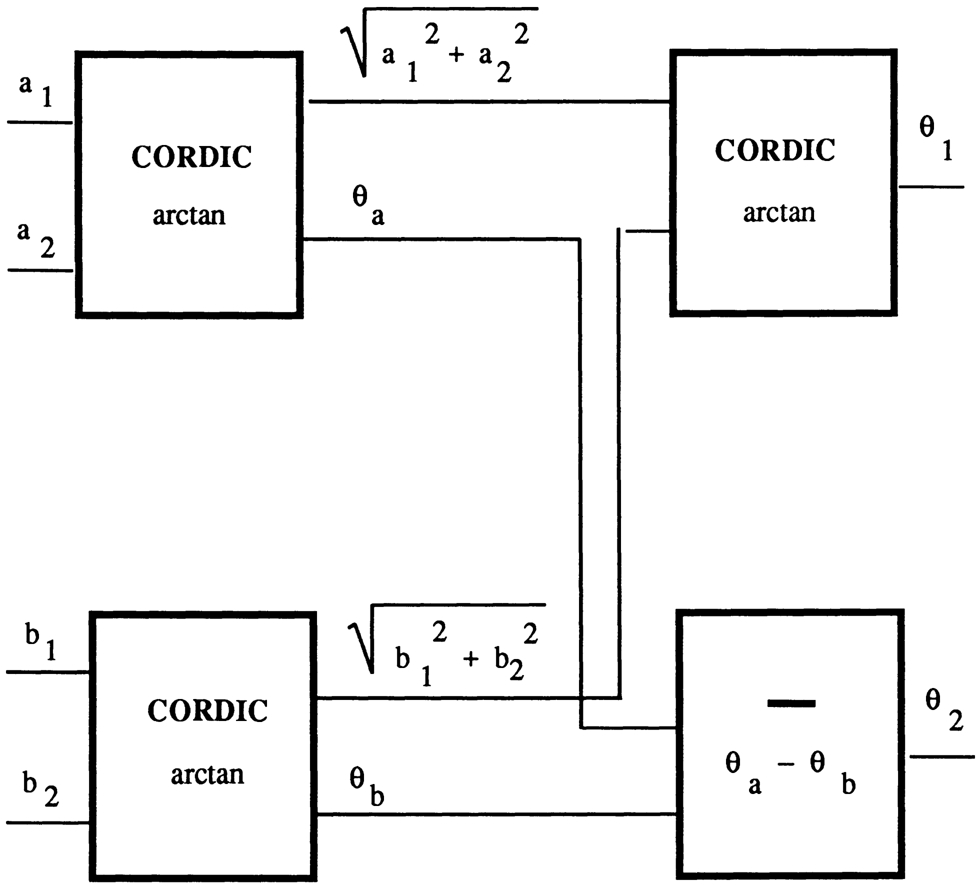


**Fig. 1. CORDIC Complex Givens Rotation Angle Calculation.**

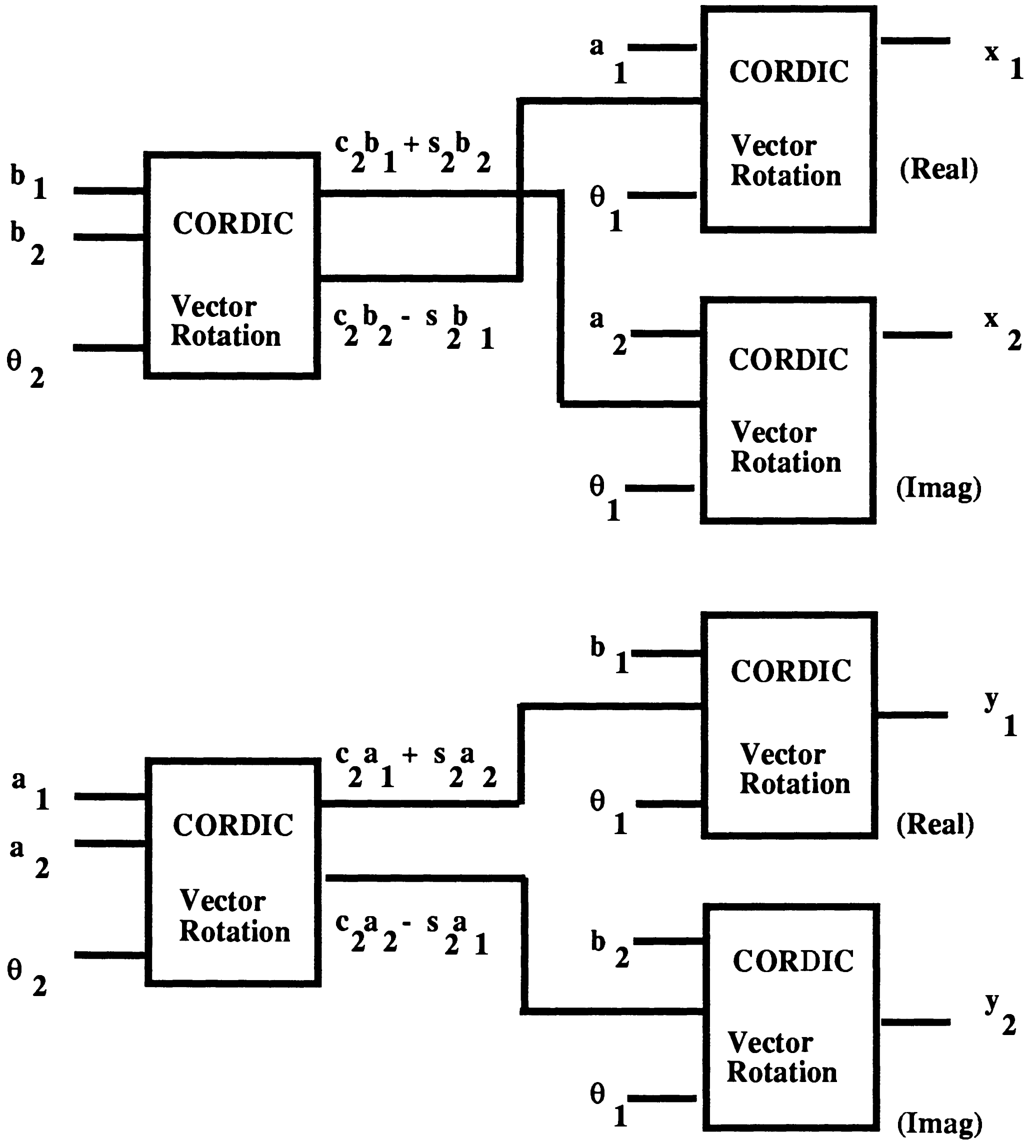


**Fig. 2. CORDIC Complex Givens Rotation Application.**
**For angle computation, see Fig. 1.**

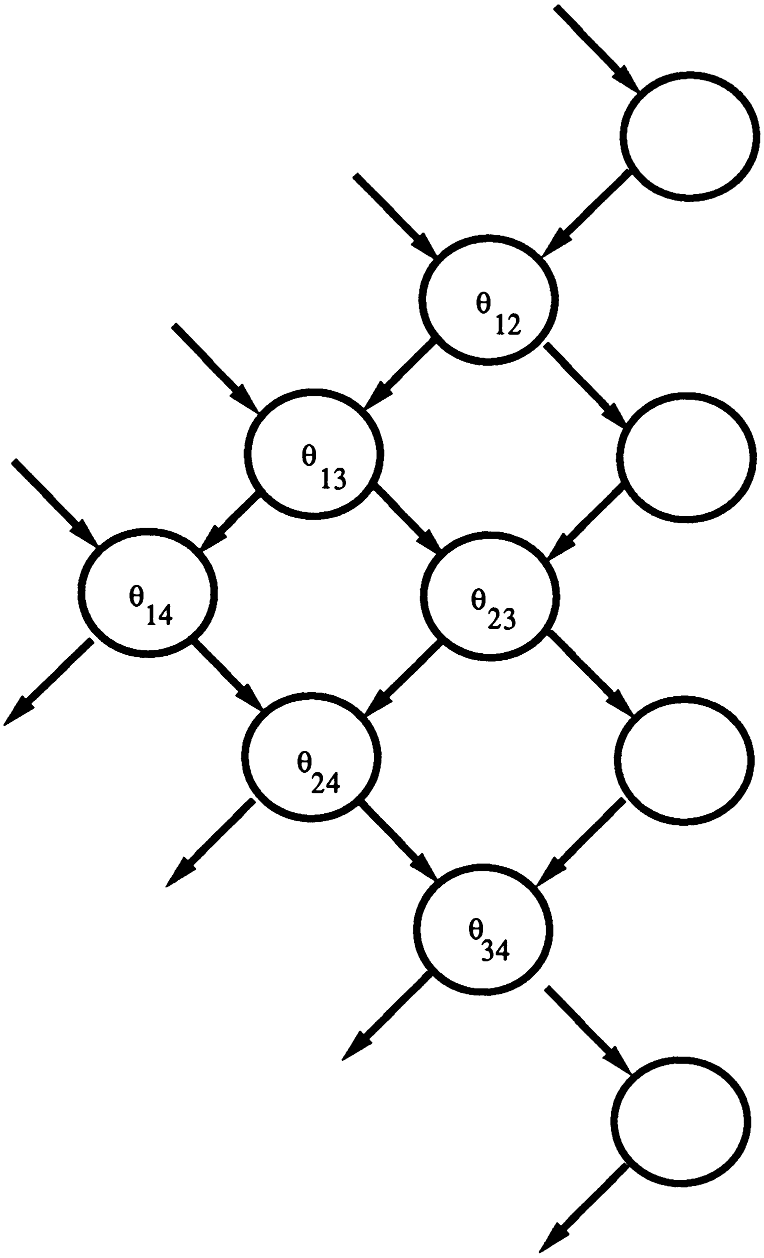


**Fig. 3. Triangular Systolic Array for the QRD.**

element. The array structure and communication remain the same. An similar array of this type has been discussed by Rader [14]. However, Rader's structure is specialized to deal with an updating procedure for the QR Decomposition. The complex CORDIC Givens procedure presented here is a more general implementation of the QR Decomposition.

# 6 Eigenvalue Decomposition of a Hermitian Matrix

The Eigenvalue Decomposition of a real symmetric matrix factors the matrix into an orthogonal matrix and a diagonal matrix of eigenvalues. The Eigenvalue Decomposition and the SVD share a similar systolic architecture due to Brent, Luk, and Van Loan [2]. This square array is shown in Figure 4. In the real Eigenvalue Decomposition, the same angle is passed from a diagonal processor along its row and column. The basic $2 \times 2$ step is

$$R(\theta)^{\mathrm{T}} \begin{bmatrix} a & b \\ c & d \end{bmatrix} R(\theta) = \begin{bmatrix} r_1 & 0 \\ 0 & r_2 \end{bmatrix} . \tag{12}$$

In the diagonal processors, the $2 \times 2$ matrix will be symmetric, that is, $b = c$. In contrast, in the SVD, the diagonal processor generates one angle which moves along the row and a different angle which mores along the column.

The eigenvalues of a Hermitian matrix may be found by a Jacobi-type algorithm. It is important to note that the Jacobi algorithm has not been successfully applied to find the eigenvalues of a general complex matrix. Other schemes, such as the Schur decomposition which uses shear transformations in addition to unitary transformations to create an upper triangular matrix have been proposed. Eberlein [7] has recently described a parallel Schur decomposition algorithm.

# 7 Singular Value Decomposition

The singular value decomposition [12] of a $p \times p$ matrix $M$ is

$$M = U \Sigma V^{\mathrm{T}} , \tag{13}$$

where $U$ and $V$ are orthogonal matrices and $\Sigma$ is a diagonal matrix of singular values. The Jacobi method for the SVD partitions a square matrix, where $p$ is even, into $2 \times 2$ submatrices. A real $2 \times 2$ SVD can be described as

$$R(\theta_l)^{\mathrm{T}} \begin{bmatrix} a & b \\ c & d \end{bmatrix} R(\theta_r) = \begin{bmatrix} \psi_1 & 0 \\ 0 & \psi_2 \end{bmatrix} , \tag{14}$$

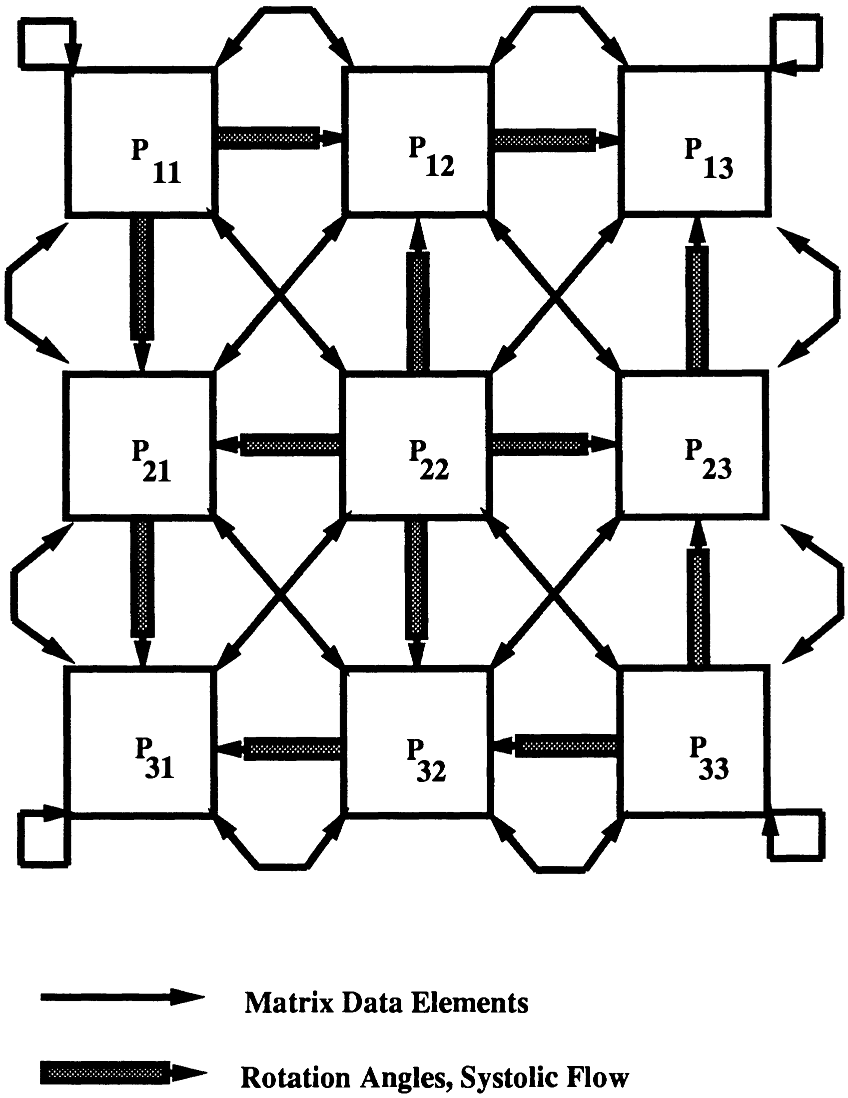


**Fig. 4. Brent-Luk-Van Loan Systolic Array for the Eigenvalue and Singular Value Decompositions.**

where $\theta_l$ and $\theta_r$ are the left and right rotation angles, respectively. The rotation matrix is

$$R(\theta) = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix}, \tag{15}$$

and the input matrix is

$$M = \begin{bmatrix} a & b \\ c & d \end{bmatrix}. \tag{16}$$

Efficient computation of the rotation parameters is essential. Several methods are possible to solve this problem. The two-step method first applies $\theta_{SYM} = (\theta_l - \theta_r)$ to symmetrize $M$ and then utilizes $\theta_r$ to diagonalize $M$. Another method, the direct two-angle approach, uses a parallel calculation of $\theta_l$ and $\theta_r$ to diagonalize $M$.

# 8 Complex SVD Algorithm

The complex SVD can be approached by using the Jacobi algorithm [10] and converting the complex 2 × 2 matrix into a real matrix. Given a complex 2 × 2 input matrix

$$M = \begin{bmatrix} a_1 + i\,a_2 & b_1 + i\,b_2 \\ c_1 + i\,c_2 & d_1 + i\,d_2 \end{bmatrix}, \tag{17}$$

several transformations are performed in order to make the 2 × 2 matrix diagonal.

First apply a complex Givens rotation [6] to introduce a zero into the 2 × 2 submatrix $M$. This is essentially a QR decomposition of $M$. The computation is as follows:

$$\begin{bmatrix} \mathrm{c}_1 & \mathrm{s}_1(\mathrm{c}_2 + i\,\mathrm{s}_2) \\ -\mathrm{s}_1(\mathrm{c}_2 - i\,\mathrm{s}_2) & \mathrm{c}_1 \end{bmatrix} \begin{bmatrix} a_1 + i\,a_2 & b_1 + i\,b_2 \\ c_1 + i\,c_2 & d_1 + i\,d_2 \end{bmatrix} = \tag{18}$$

$$\begin{bmatrix} w_1 + i\,w_2 & x_1 + i\,x_2 \\ 0 & z_1 + i\,z_2 \end{bmatrix}.$$

Then apply a unitary transformation to make the diagonal elements real:

$$U = \begin{bmatrix} e^{i\theta_1} & 0 \\ 0 & e^{i\theta_2} \end{bmatrix} \begin{bmatrix} R_w e^{i\theta_w} & R_x e^{i\theta_x} \\ 0 & R_z e^{i\theta_z} \end{bmatrix} = \tag{19}$$

$$\begin{bmatrix} W & R_x e^{i(\theta_x + \theta_1)} \\ 0 & Z \end{bmatrix} = \begin{bmatrix} W & x_1' + i\,x_2' \\ 0 & Z \end{bmatrix}.$$

Here $\theta_1 = -\theta_w$ and $\theta_2 = -\theta_z$ are the two angles which can be found using CORDIC. The entries $W$ and $Z$ are now real. The above transformation can also be considered as CORDIC vector rotations by $\theta_1$ and $\theta_2$.

Next, apply a two-sided unitary transformation to make the 2 × 2 matrix real:

$$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta_a} \end{bmatrix} \begin{bmatrix} W & R_{x'} e^{i\theta_{x'}} \\ 0 & Z \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & e^{i\theta_b} \end{bmatrix} = \begin{bmatrix} W & X \\ 0 & Z \end{bmatrix}. \tag{20}$$

where $\theta_a = \theta_{x'}$ and $\theta_b = -\theta_{x'}$.

Since the $2 \times 2$ matrix is now real, we can now proceed by performing a real $2 \times 2$ SVD. The algorithm then proceeds in a similar manner using the structure of the Brent Luk Van Loan systolic array for the Jacobi method. In the real case, only two angles are produced by each diagonal processor. However, in the complex problem presented here six additional angles are created in (17) through (20) and must be propagated through the array to update the rest of the matrix.

# 9 Complex CORDIC SVD and the Complex Array

The CORDIC algorithms are applied to the Complex SVD by building upon the complex CORDIC Givens rotation scheme. The final real SVD is then performed by using the architecture presented in Cavallaro and Luk [3]. Figure 5 shows the block diagram of a complex CORDIC SVD processor element.

The systolic array architecture of Brent, Luk, and Van Loan [2] uses an expandable square array of processors to compute the SVD of a large matrix. Figure 4 shows the systolic array structure. In the Brent, Luk, Van Loan array, the matrix is divided into $2 \times 2$ submatrices. Each processor element contains a $2 \times 2$ submatrix. The array architecture is scalable. There are two types of data flowing in this array. Rotation angles generated by the diagonal processors flow systolically along the rows and columns of the array. Matrix data elements are exchanged diagonally, after the diagonal neighbor has received and applied the necessary rotation angles. This leads to "waves" of activity moving diagonally away from the main array diagonal. Each wave is separated in time from the next by two time periods.

# 10 Area and Time Complexity of the CORDIC Arrays

In order to measure the effectiveness of the implementation, it is useful to evaluate the area and time complexity. Expressions for the area, $A_C$, and time, $T_C$, complexity of a CORDIC processor which reflect the internal arithmetic organization have previously been presented [3]. In the real Givens case, the angle calculation will require one CORDIC processor, $A_{RGA} = A_C$, and take one CORDIC time unit, $T_{RGA} = T_C$. In the complex case, the area is $A_{CGA} = 2A_C$ and the time is $T_{CGA} = 2T_C$. For the real Givens rotation application, the time is $T_{RGR} = T_C$, and the area is $A_{RGR} = 2A_C$, if we use parallel hardware to operate on each element. For the complex Givens rotation application, the

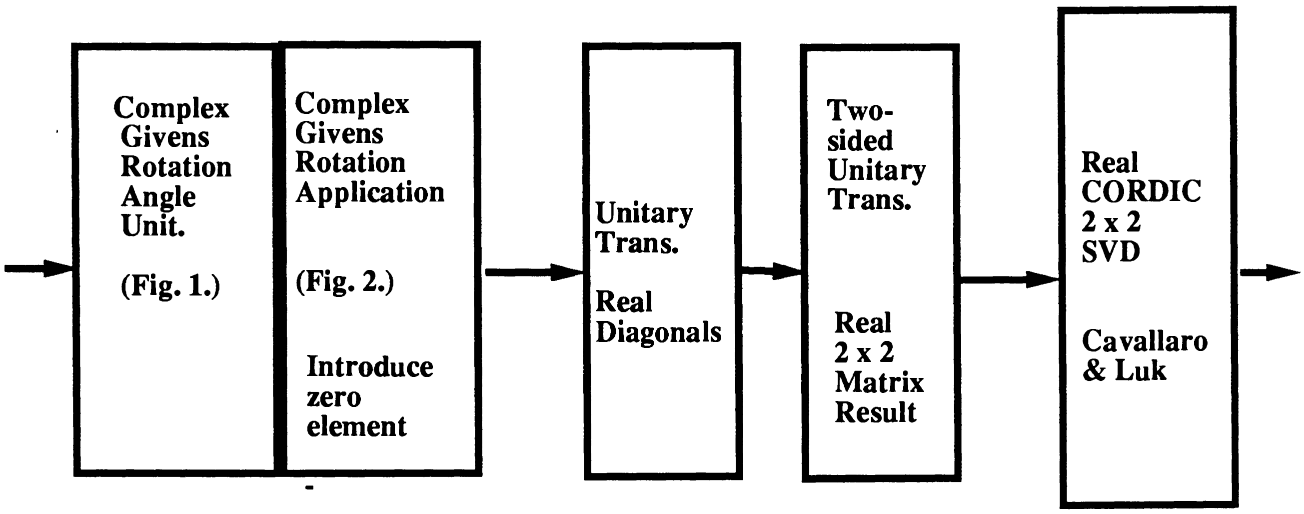


**Fig. 5. Complex CORDIC SVD Block diagram.**

time and area are:

$$T_{CGR} = 2T_C, \quad A_{CGR} = 4A_C \ , \tag{21}$$

respectively, since a CORDIC module is needed for the real and imaginary parts of each complex number.

It is possible to pipeline the angle calculation with the rotation application, since $\theta_1$ is not needed until the second step in the complex Givens rotation application. This will save $T_C$ and the area and time complexity of the total Complex Givens rotation will then be:

$$T_{CG} = 3T_C, \quad A_{CG} = 4A_C \ , \tag{22}$$

respectively. The time complexity can be further reduced if on-line CORDIC modules [9] are used.

# 11 Summary and Current Work

In this paper, the CORDIC algorithms for the real QR Decomposition and Singular Value Decomposition were extended to handle complex data. An extension of the real Givens rotation to a general *complex* Givens rotation was introduced. A CORDIC algorithm was then applied to this complex rotation. The CORDIC application was extended to the complex QRD, Eigenvalue Decomposition of a Hermitian Matrix, and the complex SVD. The area and time costs, in terms of basic CORDIC processor area and time units were presented. Currently, a VLSI implementation of a CORDIC processor array element for the real SVD is in progress at Rice University.

The analysis presented here for the complex CORDIC SVD can be extended from fixed-point to floating-point arithmetic. The complex floating-point CORDIC SVD processor would possess a hybrid structure as in the real floating-point case [4]. Many real-time signal processing applications would benefit from fault-tolerant matrix computation. Several techniques for dynamic fault reconfiguration developed for hypercubes [8] and the real CORDIC SVD array [5] can be adapted to complex matrix factorizations.

# Acknowledgments

The authors would like to thank C. F. Van Loan of Cornell University for outlining the approach for converting the complex 2 × 2 matrix into a real matrix. Thanks are also extended to C. D. Near of AT&T Bell Laboratories for providing many valuable suggestions on the SVD problem, to G. H. Golub of Stanford University for pointing out several useful references relating to the complex SVD, and to D. H. Johnson of Rice University for providing several references to the beamforming problem.

# References


[1] H. M. Ahmed, J. M. Delosme, and M. Morf. Highly Concurrent Computing Structures for Matrix Arithmetic and Signal Processing. *IEEE Computer*, 15(1):65–82, January 1982.

[2] R. P. Brent, F. T. Luk, and C. F. Van Loan. Computation of the Singular Value Decomposition Using Mesh-Connected Processors. *Journal of VLSI and Computer Systems*, 1(3):242–270, 1985.

[3] J. R. Cavallaro and F. T. Luk. CORDIC Arithmetic for an SVD Processor. *Journal of Parallel and Distributed Computing*, pages 271–290, June 1988.

[4] J. R. Cavallaro and F. T. Luk. Floating-Point CORDIC for Matrix Computations. *IEEE Int. Conf. on Computer Design*, pages 40–42, October 1988.

[5] J. R. Cavallaro, C. D. Near, and M. Ü. Uyar. Fault-Tolerant VLSI Processor Array for the SVD. *IEEE Int. Conf. on Computer Design*, pages 176–180, October 1989.

[6] T. F. Coleman and C. F. Van Loan. *Handbook for Matrix Computations.* SIAM, Philadelphia, PA, 1988.

[7] P. J. Eberlein. On the Schur Decomposition of a Matrix for Parallel Computation. *IEEE Trans. on Computers*, C-36(2):167–174, February 1987.

[8] A. C. Elster, M. Ü. Uyar, and A. P. Reeves. Fault-Tolerant Matrix Operations on Hypercube Multiprocessors. *IEEE Int'l Conference on Parallel Processing*, III:169–176, August 1989.

[9] M. D. Ercegovac and T. Lang. Singular Value Decomposition: Using Redundant and On-Line CORDIC. *Proc. SPIE Advanced Algorithms and Architectures for Signal Processing*, 975(III), August 1988.

[10] G. E. Forsythe and P. Henrici. The Cyclic Jacobi Method for Computing the Principal Values of a Complex Matrix. *Transactions of the American Mathematical Society*, 94(1):1–23, January 1960.

[11] W. M. Gentleman and H. T. Kung. Matrix Triangularization by Systolic Arrays. *Proc. SPIE Real-Time Signal Processing IV*, 298:19–26, August 1981.

[12] G. H. Golub and C. F. Van Loan. *Matrix Computations, Second Edition.* Johns Hopkins Univ. Press, Baltimore, MD, 1989.

[13] D. H. Johnson. The Application of Spectral Estimation Methods to Bearing Estimation Problems. *Proceedings of the IEEE*, 70(9):1018–1028, September 1982.

[14] C. M. Rader. Wafer-Scale Systolic Array for Adaptive Antenna Processing. *IEEE Int. Conf. on Acoustics, Speech, and Signal Processing*, pages 2069–2071, April 1988.

[15] J. Volder. The CORDIC Trigonometric Computing Technique. *IRE Trans. Electronic Computers*, EC-8(3):330–334, Sept. 1959.

[16] J. S. Walther. A Unified Algorithm for Elementary Functions. *AFIPS Spring Joint Computer Conf.*, pages 379–385, 1971.