Aleksandr CARIOW, Galina CARIOWA
WEST POMIERANIAN UNIVERSITY OF TECHNOLOGY, SZCZECIN
49 Żołnierska St., 71-210, Szczecin, Poland

Some Schemes for Implementation of Arithmetic Operations with Complex Numbers Using Squaring Units

Abstract

In this paper, new schemes for a squarer, multiplier and divider of complex numbers are proposed. Traditional structural solutions for each of these operations require the presence of some number of general-purpose binary multipliers. The advantage of our solutions is a removing of multiplications through replacing them by less costly squarers. We use Logan's trick and quarter square technique, which propose to replace the calculation of the product of two real numbers by summing the squares. Replacing usual multipliers with digital squares implies the reducing power consumption as well as decreases the complexity of the hardware circuit. The squarer requiring less area and power as compared to general-purpose multiplier, it is interesting to assess the use of squarers to implementation of complex arithmetic.

Keywords: complex number arithmetic, squaring unit, implementation complexity reduction, hardware implementation.

1. Introduction

Today, mathematical operations with complex numbers are commonly used in numerous computer science applications, such as, digital signal and image processing, telecommunication, wireless data transmission, computer games and 3D machine graphics that require data processing in real time [1-4]. Many algorithms for computing convolutions, correlations, fast orthogonal transforms, etc. require complex-valued data processing. In today's processors it is quite common to have hardware implementation of all basic operations on real-valued operands. To our knowledge, there are no implementations of complex-valued squaring, multiplication and division at the hardware level on conventional processors. With a permanent increase in capacity of integrated circuits, it is timely to consider hardware-based implementation of an extended set of operations. In complex-valued arithmetic the most time and area consuming operations are squaring, multiplication and division because these operations contain real-valued multiplications and in the latter case – real-valued divisions. In turn, multiplication and division of two real numbers are also more time and area consuming operations than addition or subtraction of real numbers. What is more, the division is even more complicated and expensive than multiplication. Thus, it can be argued that the implementation complexity of these operations is quite high. It can be shown that the complexity of the hardware implementation of the listed operations can be reduced by replacing the multipliers by the squaring units. It should be noted that squares are a special case of multiplication where both operands are identical. For this reason designers often use general-purpose multipliers to implement the squaring units by connecting multiplier's inputs together. Even though using general-purpose multipliers that are available as part of design packages reduces design time, it results in increased area and power requirements for the design [5]. Meanwhile, since the two operands are identical, some rationalizations can be made during the implementation of a dedicated squarer. In particular, unlike the general-purpose multiplier a dedicated squaring unit will have only one input, which allows to simplify the circuit. The article [6] shows that the dedicated squaring unit requires less than half whole amount of the logic gates as compared to the general-purpose multiplier. Dedicated squarer is area efficient, consumes less energy and dissipates less power as compared to general purpose multiplier. Proceeding from what has been said, we can conclude that the use of squaring units in the hardware implementation of complex-valued operations can be unusually effective.

Next, we will consider issues related to the hardware implementation of the main complex-valued operations using squaring units. Moreover, we will mean by default that we are talking about a completely parallel implementation of the proposed schemes, when each arithmetic operation is implemented by separate arithmetical unit.

2. Preliminaries

A complex number can be squared by multiplying by itself. Then in mathematical terms, such operation can be expressed as

$$z^2 = (a + jb)(a + jb) = (a^2 - b^2) + 2abj \tag{1}$$

It is well known that complex multiplication requires four real multiplications and two real additions, because:

$$(a_1 + jb_1)(a_2 + jb_2) = a_1a_2 - b_1b_2 + j(a_1b_2 + b_1a_2) \tag{2}$$

So, we can observe that the direct computation of (2) requires four real multiplications and two real additions.

By 1805 Gauss had discovered a way of reducing the number of real multiplications to three. Then, it is possible to perform the complex multiplication with three real multiplications and five real additions, because [7]:

$$\begin{aligned}(a_1 + jb_1)(a_2 + jb_2) &= a_1a_2 - b_1b_2 + \\ &+ j[(a_1 + b_1)(a_2 + b_2) - a_1a_2 - b_1b_2]\end{aligned} \tag{3}$$

To compute the quotient of two complex numbers, we multiply the numerator and denominator by the complex conjugate of the denominator:

$$\begin{aligned} z_3 = \frac{z_1}{z_2} &= \frac{a_1 + jb_1}{a_2 + jb_2} = \frac{(a_1 + jb_1)(a_2 + jb_2)}{(a_2 + jb_2)(a_2 + jb_2)} = \\ &= \frac{a_1a_2 - ja_1b_2 + ja_2b_1 - j^2b_1b_2}{a_2^2 + b_2^2} = \\ &= \frac{a_1a_2 + b_1b_2}{a_2^2 + b_2^2} + \frac{a_2b_1 - a_1b_2}{a_2^2 + b_2^2} j \end{aligned} \tag{4}$$

In 1971, Logan noted that the multiplication of two real numbers can be performed using the following expression [8]:

$$ab = \frac{1}{2}[(a + b)^2 - a^2 - b^2] \tag{4}$$

Another trick for multiplication of two real number is the so-called quarter square method [9]. The quarter square multiplication can be expressed as:

$$ab = \frac{1}{4}[(a + b)^2 - (a - b)^2] \tag{5}$$

Applying these two methods and replacing the multiplication by calculating the squares, the hardware complexity of implementing squaring operations, multiplication and division of complex numbers can be reduced. In the next part of the article such an opportunity will be shown.

3. The schemes

Let we apply Logan's trick to the calculation of the square of a complex number.

Let a_1+jb_1 is a complex number. Then the procedure for calculation of square of a complex number, represented in compact matrix notation, can be written as follows:

$$\mathbf{Y}_{2\times1} = \mathbf{A}_2\mathbf{T}_{2\times3}[\mathbf{T}_{3\times2}\mathbf{X}_{2\times1}]^2 \tag{6}$$

$$\mathbf{X}_{2\times1} = [a_1, b_1]^{\mathrm{T}},\ \mathbf{Y}_{2\times1} = [c_1, d_1]^{\mathrm{T}},$$

$$\mathbf{T}_{3\times2} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{bmatrix},\ \mathbf{T}_{2\times3} = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix},\ \mathbf{A}_2 = \begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix},$$

where $j = \sqrt{-1}$, symbol $[*]^2$ means squaring all the entries of the vector inscribed inside of the square brackets, a_1 - is real part of complex number, b_1 - is an imaginary part of complex number, c_1- is real part of square of complex number, and d_1- is an imaginary part of square of complex number.

Fig. 1 shows a scheme for calculating the square of a complex number using squaring units. In this paper, all schemes are oriented from left to right. Straight lines in the schemes denote the operations of data transfer. The circles in these schemes show the operation of multiplication by a number inscribed inside the circle. The small squares in this figure indicate the squaring operations. Points where lines converge denote summation. We use the usual lines without arrows on purpose, so as not to clutter the scheme.

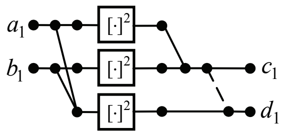

Fig. 1. Diagram for calculating the square of a complex number using squaring units

Now we apply the Gauss's trick and the quarter square method to the calculation of the product of two complex numbers.

Let a_1+jb_1 is a complex multiplier, a_2+jb_2 is a complex multiplicand and c_1+jd_1 is a complex number product. Then the computation procedure for complex number multiplication, represented in matrix notation, can be written as follows:

$$\mathbf{Y}_{2\times1} = \mathbf{D}_2\mathbf{A}_{2\times3}\mathbf{A}_{3\times6}[\hat{\mathbf{H}}_6\mathbf{A}_{6\times4}\mathbf{X}_{4\times1}]^2 \tag{7}$$

$$\mathbf{A}_{6\times4} = \left[\begin{array}{c:c:c:c} 0 & 0 & 1 & -1 \\ \hdashline 1 & 0 & 0 & 0 \\ \hdashline 0 & 0 & 1 & 1 \\ \hdashline 0 & 1 & 0 & 0 \\ \hdashline 0 & 0 & 0 & 1 \\ \hdashline 1 & -1 & 0 & 0 \end{array}\right],\ \mathbf{A}_{3\times6} = \left[\begin{array}{cc:cc:cc} 1 & -1 & 0 & 0 & 0 & 0 \\ \hdashline 0 & 0 & 1 & -1 & 0 & 0 \\ \hdashline 0 & 0 & 0 & 0 & 1 & -1 \end{array}\right],$$

$$\hat{\mathbf{H}}_6 = \mathbf{I}_3 \otimes \mathbf{H}_2 = \left[\begin{array}{cc:cc:cc} 1 & 1 & & & & \\ 1 & -1 & & & & \\ \hdashline & & 1 & 1 & & \\ & & 1 & -1 & & \\ \hdashline & & & & 1 & 1 \\ & & & & 1 & -1 \end{array}\right],\ \mathbf{I}_3 = \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix},$$

$$\mathbf{H}_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix},\ \mathbf{A}_{2\times3} = \begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix},\ \mathbf{D}_2 = diag(1/4, 1/4),$$

$$\mathbf{X}_{4\times1} = [a_1, b_1, a_2, b_2]^{\mathrm{T}}$$

where „$\otimes$" denotes the Kronecker product of two matrices [10], a_1, a_2 - are real parts of complex multiplier and multiplicand, b_1, b_2 - are imaginary parts of complex multiplier and multiplicand, and c_2- is real part and d_2 is an imaginary part of complex number product respectively.

Fig. 2 shows a scheme for multiplying two complex numbers using squaring units. The rectangles indicate the operations of multiplication by the matrices $\mathbf{H}_2$.

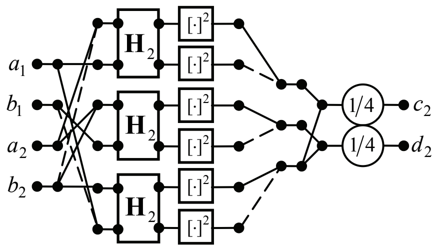

Fig. 2. Diagram for complex-valued multiplication using squaring units

Finally, we apply Logan's trick to the calculation of quotient of two complex numbers. Let a_1+jb_1 be a complex dividend, a_2+jb_2 be a complex divider and c_3+jd_3 be a quotient of two complex numbers. Then the corresponding procedure, represented in matrix form, can be written as follows:

$$\mathbf{Y}_{2\times1} = \hat{\mathbf{D}}_2\hat{\mathbf{A}}_{2\times3}\mathbf{A}_{3\times4}\mathbf{A}_{4\times8}[\mathbf{A}_8\mathbf{P}_{8\times4}\mathbf{X}_{4\times1}]^2 \tag{8}$$

$$\mathbf{X}_{4\times1} = [a_1, a_2, b_1, b_2]^{\mathrm{T}},\ \mathbf{P}_{8\times4} = \mathbf{1}_2 \otimes \mathbf{I}_4 = \left[\begin{array}{cccc} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \\ \hdashline 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{array}\right],$$

$$\mathbf{A}_8 = \left[\begin{array}{cccc:cccc} 1 & 1 & 0 & 0 & & & & \\ 0 & 0 & 1 & 1 & & \mathbf{0}_4 & & \\ 0 & 1 & 1 & 0 & & & & \\ 1 & 0 & 0 & 1 & & & & \\ \hdashline & & & & 1 & 0 & 0 & 0 \\ & \mathbf{0}_4 & & & 0 & 0 & 1 & 0 \\ & & & & 0 & 1 & 0 & 0 \\ & & & & 0 & 0 & 0 & 1 \end{array}\right],$$

$$\mathbf{A}_{4\times8} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix},$$

$$\mathbf{A}_{3\times4} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix},\ \hat{\mathbf{A}}_{2\times3} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 1 & -1 \end{bmatrix},$$

$$\hat{\mathbf{D}}_2 = \mathbf{I}_2 \otimes 1/2(a_2^2 + b_2^2)$$

and $\mathbf{0}_{N\times M}$ is an $M\times N$ matrix of zeros (a matrix where every element is equal to zero), $\mathbf{I}_N$ - is an identity matrix, $\mathbf{1}_2 = [1,1]^{\mathrm{T}}$ [11].

Fig. 3 shows a scheme for calculation of quotient of two complex numbers using squaring units.

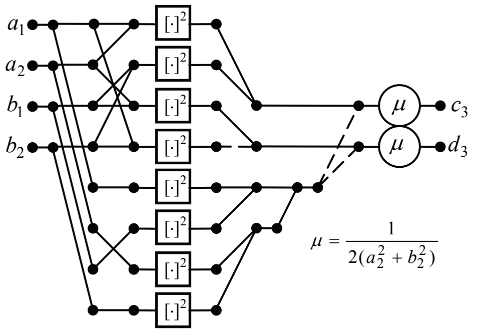

Fig. 3. Diagram for complex-valued division using squaring units

4. Implementation complexity

Tables 1 and 2 show the nomenclature and the number of arithmetic units that are necessary for the hardware implementation of the described complex-valued operations in the case of using the traditional arithmetic units and in the case of using the squaring units, respectively. Analyzing these data, it is easy to verify that in some cases the hardware implementation of the proposed schemes is more efficient than the hardware implementation using traditional arithmetic blocks.

Tab. 1. The nomenclature and the number of arithmetic units necessary for the implementation of complex-valued operations using direct methods

Type of arithmetic unit	Type of implemented complex-valued operation		
	squaring	multiplication	division
Adders	1	2	3
Squarers	2	—	2
Multipliers	1	4	4
Dividers	—	—	2

Tab. 2. The nomenclature and the number of arithmetic units necessary for the implementation of complex-valued operations using the proposed schemes

Type of arithmetic unit	Type of implemented complex-valued operation		
	squaring	multiplication	division
Adders	3	14	11
Squarers	3	6	8
Multipliers	—	—	—
Dividers	—	—	2

5. Conclusions

The article presents three new schemes for fully parallel hardware implementation of basic complex arithmetic operations, namely: squaring, multiplication and division using squaring units. To reduce the hardware complexity (number of two-operand multipliers), we exploit the Logan's identity and quarter square method for replacing the binary multipliers by the squaring units. If the requirements for the speed of solving the task are not high, then in order to minimize the chip area and reducing power consumption, only one squaring unit can be reused many times. We provide an opportunity for an inquisitive reader to make sure of the effectiveness of the proposed solutions independently.

6. References

[1] Nikolova Z., Iliev G., Ovtcharov M. and Poulkov V.: Complex Digital Signal Processing in Telecommunications. In "Applications of Digital Signal Processing" Edited by Christian Cuadrado-Laborde, InTech Open Access Publisher, 2011, pp. 3-24.

[2] Brown J. A., Crowder H.: Graphics Application Using Complex Numbers in APL2 Technical Report: TR 03.265, Santa Teresa Laboratory San Jose, CA. March 1985

[3] Martin, K.: Complex signal processing is not – complex. Proc. of the 29th European Conf. on Solid-State Circuits (ESSCIRC'03), 2003, pp. 3-14, Estoril, Portugal, 16-18 Sept. 2003.

[4] Cariow A., Cariowa G.: An algorithm for complex-valued vector-matrix multiplication. Electrical Review, 2012, R88, No 10b, pp. 213-216.

[5] Deshpande A., Draper J.: Squaring units and a comparison with multipliers. In 53rd IEEE International Midwest Symp. on Circuits and Systems (MWSCAS 2010), Seattle, Washington August 1st-4th, 2010, pp. 1266 – 1269, doi: 10.1109/MWSCAS.2010.5548763.

[6] Liddicoat A. A., Flynn M. J.: Parallel Computation of the Square and Cube Function. Computer Systems Laboratory; Stanford University, Technical report No. CSL-TR-00-808, August 2000.

[7] Knuth D.E.: The Art Of Computing Programming, Volume 2, Seminumerical Algorithms, Addison-Wesley, Reading, MA, USA, Second Ed., 1981.

[8] Logan J.R.: A square-summing high-speed multiplier. Comput. Des., 1971, pp. 67-70.

[9] Johnson E. L.: A Digital Quarter Square Multiplier. IEEE Transactions on Computers, vol. C-29, no. 3, pp. 258 – 261, 1980, doi:10.1109/TC.1980.1675558.

[10] Granata J., Conner M., Tolimieri R.: The Tensor product: A mathematical programming language for FFTs and other fast DSP operations. IEEE Signal Processing Magazine, 1992. No 1, pp.41-48.

[11] Cariow A.: Strategies for the Synthesis of Fast Algorithms for the Computation of the Matrix-vector Products. Journal of Signal Processing Theory and Applications, 2014, vol. 3 No. 1 pp. 1-19.

Received: 21.03.2017 *Paper reviewed* *Accepted: 03.05.2017*

Prof. Aleksandr CARIOW, DSc.

He received the Candidate of Sciences (PhD) and Doctor of Sciences degree (DSc or Habilitation) in Computer Sciences from LITMO of St. Petersburg, Russia in 1984 and 2001, respectively. In September 1999, he joined the faculty of Computer Sciences and Information Technology at the West Pomeranian University of Technology, Szczecin, Poland, where he is currently a professor and chair of the Department of Computer Architectures and Telecommunications. His research interests include digital signal processing algorithms, VLSI architectures, data processing parallelization.

e-mail: acariow@wi.zut.edu.pl

Galina CARIOWA, PhD

She received the MSc degree in Mathematics from Moldavian State University, Chişinău in 1978, and PhD degree in computer science from West Pomeranian University of Technology, Szczecin, Poland in 2007. She is currently working as an assistant professor of the Department of Multimedia Systems. She is also an Associate-Editor of World Research Journal of Transactions on Algorithms. Her scientific interests include numerical linear algebra and digital signal processing algorithms, VLSI architectures, and data processing parallelization.

e-mail: gcariowa@wi.zut.edu.pl