Information Processing Letters 113 (2013) 324–331

Contents lists available at SciVerse ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

An algorithm for fast multiplication of sedenions

Aleksandr Cariow [*], Galina Cariowa

West Pomeranian University of Technology, Szczecin, Department of Computer Architectures and Telecommunications, ul. Zolnierska 49, 71-210 Szczecin, Poland

ARTICLE INFO

Article history:
Received 18 September 2012
Received in revised form 21 February 2013
Accepted 22 February 2013
Available online 26 February 2013
Communicated by B. Doerr

Keywords:
Design of algorithms
Sedenion
Multiplication

ABSTRACT

In this work a rationalized algorithm for calculating the product of sedenions is presented which reduces the number of underlying multiplications. Therefore, reducing the number of multiplications in VLSI processor design is usually a desirable task. The computation of a sedenion product using the naive method takes 256 multiplications and 240 additions, while the proposed algorithm can compute the same result in only 122 multiplications (or multipliers – in hardware implementation case) and 298 additions.

1. Introduction

Algebras of hypercomplex numbers [1] are widely used in various fields of data processing including digital signal and image processing [2–5] and telecommunications [6]. For hypercomplex algebras the most time-consuming operation is the multiplication of two hypercomplex numbers. The schoolbook multiplication of complex numbers require performing four real multiplications, the multiplication of quaternions requires performing 16 real multiplications, and the multiplication of octonions and sedenions require performing 64 and 256 real multiplications, respectively. We see that the growth of the algebra's dimension leads to an increase in the number of multiplications of real numbers needed to calculate hypercomplex number products. It would be nice if when calculating the product of hypercomplex numbers, the number of real multiplications needed for this purpose could have been reduced.

Efficient algorithms for the multiplication of quaternions, biquaternions and octonions exist [7–11]. No such algorithms for the multiplication of sedenions have been proposed. The aim of the present paper is to suggest an efficient algorithm for this purpose.

A sedenion is defined as follows [12]

$$S = s_0 + \sum_{i=1}^{15} s_i e_i$$

where s_0 and $\{s_i\}$, $i = 1, \dots, 15$, are real numbers and $\{e_i\}$, $i = 1, \dots, 15$, are imaginary units.

The following notations are used:

$$A = a_0 + \sum_{i=1}^{15} a_i e_i, \qquad B = b_0 + \sum_{i=1}^{15} b_i e_i,$$

$$C = AB = c_0 + \sum_{i=1}^{15} c_i e_i.$$

The sedenions product can be presented in the matrix–vector multiplication form as [13]

$$\mathbf{C}_{16\times 1} = \mathbf{B}_{16}\mathbf{A}_{16\times 1} \tag{1}$$

where

$$\mathbf{A}_{16\times 1} = [a_0, a_1, \dots, a_{15}]^{\mathrm{T}},$$

$$\mathbf{C}_{16\times 1} = [c_0, c_1, \dots, c_{15}]^{\mathrm{T}},$$

[*] Corresponding author. Tel.: +48 449 55 73.
E-mail addresses: atariov@wi.zut.edu.pl (A. Cariow), gtariova@wi.zut.edu.pl (G. Cariowa).

0020-0190/$ – see front matter Crown Copyright © 2013 Published by Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.ipl.2013.02.011

$$\mathbf{B}_{16} = \left[\begin{array}{c|c|c|c} \mathbf{B}_4^{(0,0)} & \mathbf{B}_4^{(0,1)} & \mathbf{B}_4^{(0,2)} & \mathbf{B}_4^{(0,3)} \\ \hline \mathbf{B}_4^{(1,0)} & \mathbf{B}_4^{(1,1)} & \mathbf{B}_4^{(1,2)} & \mathbf{B}_4^{(1,3)} \\ \hline \mathbf{B}_4^{(2,0)} & \mathbf{B}_4^{(2,1)} & \mathbf{B}_4^{(2,2)} & \mathbf{B}_4^{(2,3)} \\ \hline \mathbf{B}_4^{(3,0)} & \mathbf{B}_4^{(3,1)} & \mathbf{B}_4^{(3,2)} & \mathbf{B}_4^{(3,3)} \end{array}\right]$$

and

$$\mathbf{B}_4^{(0,0)} = \begin{bmatrix} b_0 & -b_1 & -b_2 & -b_3 \\ b_1 & b_0 & b_3 & -b_2 \\ b_2 & -b_3 & b_0 & b_1 \\ b_3 & b_2 & -b_1 & b_0 \end{bmatrix},$$

$$\mathbf{B}_4^{(0,1)} = \begin{bmatrix} -b_4 & -b_5 & -b_6 & -b_7 \\ b_5 & -b_4 & -b_7 & b_6 \\ b_6 & b_7 & -b_4 & -b_5 \\ b_7 & -b_6 & b_5 & -b_4 \end{bmatrix},$$

$$\mathbf{B}_4^{(0,2)} = \begin{bmatrix} -b_8 & -b_9 & -b_{10} & -b_{11} \\ b_9 & -b_8 & -b_{11} & b_{10} \\ b_{10} & b_{11} & -b_8 & -b_9 \\ b_{11} & -b_{10} & b_9 & -b_8 \end{bmatrix},$$

$$\mathbf{B}_4^{(0,3)} = \begin{bmatrix} -b_{12} & -b_{13} & -b_{14} & -b_{15} \\ -b_{13} & b_{12} & b_{15} & -b_{14} \\ -b_{14} & -b_{15} & b_{12} & b_{13} \\ -b_{15} & b_{14} & -b_{13} & b_{12} \end{bmatrix},$$

$$\mathbf{B}_4^{(1,0)} = \begin{bmatrix} b_4 & -b_5 & -b_6 & -b_7 \\ b_5 & b_4 & -b_7 & b_6 \\ b_6 & b_7 & b_4 & -b_5 \\ b_7 & -b_6 & b_5 & b_4 \end{bmatrix},$$

$$\mathbf{B}_4^{(1,1)} = \left[\mathbf{B}_4^{(0,0)}\right]^{\mathrm{T}} = \begin{bmatrix} b_0 & b_1 & b_2 & b_3 \\ -b_1 & b_0 & -b_3 & b_2 \\ -b_2 & b_3 & b_0 & -b_1 \\ -b_3 & -b_2 & b_1 & b_0 \end{bmatrix},$$

$$\mathbf{B}_4^{(1,2)} = \left[-\mathbf{B}_4^{(0,3)}\right]^{\mathrm{T}} = \begin{bmatrix} b_{12} & b_{13} & b_{14} & b_{15} \\ b_{13} & -b_{12} & b_{15} & -b_{14} \\ b_{14} & -b_{15} & -b_{12} & b_{13} \\ b_{15} & b_{14} & -b_{13} & -b_{12} \end{bmatrix},$$

$$\mathbf{B}_4^{(1,3)} = \begin{bmatrix} -b_8 & -b_9 & -b_{10} & -b_{11} \\ b_9 & -b_8 & b_{11} & -b_{10} \\ b_{10} & -b_{11} & -b_8 & b_9 \\ b_{11} & b_{10} & -b_9 & -b_8 \end{bmatrix},$$

$$\mathbf{B}_4^{(2,0)} = \left[-\mathbf{B}_4^{(0,2)}\right]^{\mathrm{T}} = \begin{bmatrix} b_8 & -b_9 & -b_{10} & -b_{11} \\ b_9 & b_8 & -b_{11} & b_{10} \\ b_{10} & b_{11} & b_8 & -b_9 \\ b_{11} & -b_{10} & b_9 & b_8 \end{bmatrix},$$

$$\mathbf{B}_4^{(2,1)} = \mathbf{B}_4^{(0,3)} = \begin{bmatrix} -b_{12} & -b_{13} & -b_{14} & -b_{15} \\ -b_{13} & b_{12} & b_{15} & -b_{14} \\ -b_{14} & -b_{15} & b_{12} & b_{13} \\ -b_{15} & b_{14} & -b_{13} & b_{12} \end{bmatrix},$$

$$\mathbf{B}_4^{(2,2)} = \mathbf{B}_4^{(1,1)} = \begin{bmatrix} b_0 & b_1 & b_2 & b_3 \\ -b_1 & b_0 & -b_3 & b_2 \\ -b_2 & b_3 & b_0 & -b_1 \\ -b_3 & -b_2 & b_1 & b_0 \end{bmatrix},$$

$$\mathbf{B}_4^{(2,3)} = \begin{bmatrix} b_4 & b_5 & b_6 & b_7 \\ -b_5 & b_4 & b_7 & -b_6 \\ -b_6 & -b_7 & b_4 & b_5 \\ -b_7 & b_6 & -b_5 & b_4 \end{bmatrix},$$

$$\mathbf{B}_4^{(3,0)} = \left[-\mathbf{B}_4^{(2,1)}\right]^{\mathrm{T}} = \begin{bmatrix} b_{12} & b_{13} & b_{14} & b_{15} \\ b_{13} & -b_{12} & b_{15} & -b_{14} \\ b_{14} & -b_{15} & -b_{12} & b_{13} \\ b_{15} & b_{14} & -b_{13} & -b_{12} \end{bmatrix},$$

$$\mathbf{B}_4^{(3,1)} = \left[-\mathbf{B}_4^{(1,3)}\right]^{\mathrm{T}} = \begin{bmatrix} b_8 & -b_9 & -b_{10} & -b_{11} \\ b_9 & b_8 & b_{11} & -b_{10} \\ b_{10} & -b_{11} & b_8 & b_9 \\ b_{11} & b_{10} & -b_9 & b_8 \end{bmatrix},$$

$$\mathbf{B}_4^{(3,2)} = \left[\mathbf{B}_4^{(0,1)}\right]^{\mathrm{T}} = \begin{bmatrix} -b_4 & b_5 & b_6 & b_7 \\ -b_5 & -b_4 & b_7 & -b_6 \\ -b_6 & -b_7 & -b_4 & b_5 \\ -b_7 & b_6 & -b_5 & -b_4 \end{bmatrix},$$

$$\mathbf{B}_4^{(3,3)} = \mathbf{B}_4^{(0,0)} = \begin{bmatrix} b_0 & -b_1 & -b_2 & -b_3 \\ b_1 & b_0 & b_3 & -b_2 \\ b_2 & -b_3 & b_0 & b_1 \\ b_3 & b_2 & -b_1 & b_0 \end{bmatrix}.$$

The direct multiplication of the vector–matrix product in Eq. (1) requires 256 real multiplications and 240 additions. We present an algorithm, which reduces multiplicative complexity to 122 multiplications at the price of 58 more additions.

2. Synthesis of a fast algorithm for sedenion multiplication

The idea is that the original sedenion multiplication matrix $\mathbf{B}_{16}$ can be decomposed as an algebraic sum of a symmetric Toeplitz matrix and another matrix which has many zero elements. The Toeplitz matrix is shift-structured and a number of algorithms exist for fast matrix–vector multiplication. For instance, the matrix can be diagonalized using the Fast Hadamard transform (FHT) and thus vector–matrix products can be computed efficiently. The same idea was originally used for the multiplication of quaternions in [9]. We have extended this idea in our previous paper [10] for the case of octonions. In this article we further extend this idea to the case of the sedenions.

Let $\mathbf{Y}_{16\times 1} = [-c_0, c_1, c_2, \ldots, c_{15}]^{\mathrm{T}}$ and $\mathbf{X}_{16\times 1} = [a_0, a_1, \ldots, a_{15}]^{\mathrm{T}}$.

Let us multiply the first row of $\mathbf{B}_{16}$ by (-1). (We can easily see that this transformation leads in the future to minimize the computational complexity of the final algorithm.) This transformation is done in order to present a modified in this manner matrix as an algebraic sum of the block-symmetric Toeplitz-type matrix and some sparse matrix, i.e. matrix containing only small number of non-zero elements.

Then we can write

$$\mathbf{Y}_{16\times 1} = \breve{\mathbf{B}}_{16}\mathbf{X}_{16\times 1} - 2\hat{\mathbf{B}}_{16}\mathbf{X}_{16\times 1} \tag{2}$$

where

$$\breve{\mathbf{B}}_{16} = \left[\begin{array}{c|c|c|c} \breve{\mathbf{B}}_4^{(0,0)} & \breve{\mathbf{B}}_4^{(0,1)} & \breve{\mathbf{B}}_4^{(0,2)} & \breve{\mathbf{B}}_4^{(0,3)} \\ \hline \breve{\mathbf{B}}_4^{(0,1)} & \breve{\mathbf{B}}_4^{(0,0)} & \breve{\mathbf{B}}_4^{(0,3)} & \breve{\mathbf{B}}_4^{(0,2)} \\ \hline \breve{\mathbf{B}}_4^{(0,2)} & \breve{\mathbf{B}}_4^{(0,3)} & \breve{\mathbf{B}}_4^{(0,0)} & \breve{\mathbf{B}}_4^{(0,1)} \\ \hline \breve{\mathbf{B}}_4^{(0,3)} & \breve{\mathbf{B}}_4^{(0,2)} & \breve{\mathbf{B}}_4^{(0,1)} & \breve{\mathbf{B}}_4^{(0,0)} \end{array}\right],$$