Aleksandr CARIOW [1,2], Wojciech SYSŁO [2], Galina CARIOWA [1], Marek GLISZCZYŃSKI [1]
[1] WEST POMIERANIAN UNIVERSITY OF TECHNOLOGY, SZCZECIN, Żołnierska St. 49, 71-210 Szczecin
[2] HIGHER VOCATION STATE SCHOOL IN GORZÓW WIELKOPOLSKI, Teatralna St. 25, 66-400 Gorzów Wlkp.

A rationalized structure of processing unit to multiply 3×3 matrices

Dr hab. inż Aleksandr CARIOW

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

e-mail: atariov@wi.ps.pl

Dr Galina CARIOWA

She received the MSc degrees in mathematics from Moldavian State University, Chişinău in 1976 and PhD degree in computer science from West Pomeranian University of Technology, Szczecin, Poland in 2007. She is currently working as assistant professor of the Department of Computer Architectures and Telecommunications. Her scientific interests include digital signal processing algorithms, VLSI architectures, and data processing parallelization.

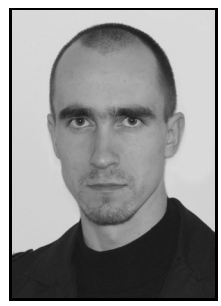
e-mail: atariov@wi.ps.pl

Dr Wojciech SYSŁO

Wojciech A. Sysło obtained his PhD degree in experimental physics from the University of Wrocław in 1979. Currently, he is a vice director of the Technical Institute of Higher Vocational State School in Gorzów Wielkopolski. His current research interests are in education of information technology, and technical education.

e-mail: syslow@o2.pl

Mgr inż. Marek GLISZCZYŃSKI



He was born in Lębork, Poland in 1985. He received the M.S. degree in Computer Science and Information Engineering in 2009 from West Pomeranian University of Technology, Szczecin. He is currently a PhD student in the Faculty of Computer Sciences and Informatics Technologies in West Pomeranian University of Technology. His research interests include digital signal processing algorithms, VLSI architectures, and data processing parallelization.

e-mail: mgliszczyński@wi.ps.pl

Abstract

This paper presents a high-speed parallel 3x3 matrix multiplier structure. To reduce the hardware complexity of the multiplier structure, we propose to modify the Makarov's algorithm for 3×3 by 3×3 matrix multiplication. The process of matrix product calculation is successively decomposed so that a minimal set of multipliers and fewer adders are used to generate partial results which are combined to generate the final results. Thus, our proposed modification reduces the number of adders compared to the direct implementation of the Makarov's algorithm, and takes advantage of parallelism of calculation offered by field-programmable gate arrays (FPGA's).

Keywords: matrix multiplier, hardware complexity reduction, (FPGA) implementation.

Zracjonalizowana struktura jednostki procesorowej do mnożenia macierzy trzeciego stopnia

Streszczenie

W pracy została przedstawiona struktura jednostki procesorowej do wyznaczania iloczynu dwóch macierzy trzeciego stopnia. W odróżnieniu od implementacji naiwnego sposobu zrównoleglenia obliczeń wymagającego 27 układów mnożących proponowana równoległa struktura wymaga tylko 22 układa mnożących. A ponieważ układ mnożący pochłania znacznie więcej zasobów sprzętowych platformy implementacyjnej niż sumator, to minimalizacja układów mnożących przy projektowaniu mikroelektronicznych jednostek procesorowych jest sprawą nadrzędną. Zasada budowy proponowanej jednostki oparta jest na realizacji autorskiej modyfikacji metody Makarova, z tym, że implementacja naszej modyfikacji wymaga o 38 sumatorów mniej niż implementacja metody Makarova. Zaproponowana struktura może bycz z powodzeniem zastosowana do akceleracji obliczeń w podsystemach cyfrowego przetwarzania danych zrealizowanych na platformach FPGA oraz zaimplementowana w dowolnym środowisku sprzętowym, na przykład zrealizowana w postaci układu ASIC. W tym ostatnim przypadku niewątpliwym atutem wyróżniającym przedstawione rozwiązanie jest to, że zaprojektowany w ten sposób układ będzie zużywać mniej energii oraz wydzielać mniej ciepła.

Słowa kluczowe: układ mnożenia macierzy, redukcja złożoności sprzętowej, implementacja na FPGA.

1. Introduction

The order of the multiplicative complexity is commonly used to measure and compare the efficiency of the algorithms since multiplications are intrinsically more complicated among all operations.

Since the multiplier requires considerably more hardware resources than the adder, a smaller number of multipliers consumes less power. Therefore reducing the number of multipliers in processor unit structures is usually a desirable task [1-3].

The objective, in most cases, is the design of computational algorithms and their respective implementation in a manner performing the required computations in the least amount of time. In order to achieve this goal, parallel processing has also received a lot of attention in the research community [4].

Matrix multiplication (MM) is one of the most widely used scientific computer tasks. The number of applications of MM continues to grow and includes such diverse areas as: signal processing, communications, computer graphics, and biometrics.

2. Synthesis of new version of Makarov's method for 3×3 matrix product computing

Let $\mathbf{X}_3$ and $\mathbf{Y}_3$ be two square matrices of size 3×3. The problem is to find the matrix

$$\mathbf{Z}_3 = \mathbf{Y}_3 \mathbf{X}_3 , \tag{1}$$

where

$$\mathbf{Z}_3 = \begin{bmatrix} z_{0,0} & z_{0,1} & z_{0,2} \\ z_{1,0} & z_{1,1} & z_{1,2} \\ z_{2,0} & z_{2,1} & z_{2,2} \end{bmatrix},$$

$$\mathbf{X}_3 = \begin{bmatrix} x_{0,0} & x_{0,1} & x_{0,2} \\ x_{1,0} & x_{1,1} & x_{1,2} \\ x_{2,0} & x_{2,1} & x_{2,2} \end{bmatrix},$$

$$\mathbf{Y}_3 = \begin{bmatrix} y_{0,0} & y_{0,1} & y_{0,2} \\ y_{1,0} & y_{1,1} & y_{1,2} \\ x_{y2,0} & y_{2,1} & y_{2,2} \end{bmatrix}$$

Direct and full parallel implementation of matrix product (1) requires 27 multipliers and 18 adders, however by exploiting some rationalization solutions the number of multipliers could be decreases. Many effective realizations of this algorithm have been reported in the literature [5-10], but analysis of these approaches shows that possibilities for developing 3×3 MM algorithms are not exhausted. In the present work a new variant of Makarov's method realization is presented.

Let

$$M_1 = (y_{0,2} + L_1)(x_{0,0} + M_1),$$
$$M_2 = (y_{0,1} + L_2)(x_{1,0} - N_2),$$
$$M_3 = (y_{0,1} + L_3)(x_{2,0} - N_2),$$
$$M_4 = (y_{0,2} - L_4)(x_{2,0} - N_1),$$
$$M_5 = (y_{0,0} - L_1)x_{0,0},$$
$$M_6 = (y_{0,0} + L_1)x_{1,0},$$
$$M_7 = (y_{0,0} + L_3 + L_4)x_{2,0},$$
$$M_8 = y_{0,1}(x_{0,0} + N_2),$$
$$M_9 = y_{0,2}(x_{1,0} + N_1),$$
$$M_{10} = x_{0,1}y_{1,0},$$
$$M_{11} = x_{1,2}y_{2,0},$$
$$M_{12} = L_1N_3,$$
$$M_{13} = L_2N_4,$$
$$M_{14} = L_5N_2,$$
$$M_{15} = x_{1,1}y_{1,2},$$
$$M_{16} = L_6N_1$$
$$M_{17} = x_{1,2}y_{2,1},$$
$$M_{18} = (x_{2,0} - L_4)y_{1,2},$$
$$M_{19} = (L_7 - L_3)y_{2,1},$$
$$M_{20} = L_3(N_5 + N_6),$$
$$M_{21} = L_4(N_7 - N_6),$$
$$M_{22} = (L_4 - L_3)N_6,$$

$$L_1 = x_{0,2} - x_{1,2},$$
$$L_2 = x_{0,1} + x_{1,1},$$
$$L_3 = x_{0,1} + x_{2,0},$$
$$L_4 = x_{1,2} + x_{2,2},$$
$$L_5 = y_{0,1} + x_{0,1},$$
$$L_6 = y_{0,2} - x_{1,2},$$
$$L_7 = x_{0,2} + x_{2,2},$$

$$N_1 = y_{2,0} - y_{2,1} + y_{2,2},$$
$$N_2 = y_{2,0} - y_{2,1} + y_{2,2},$$
$$N_3 = x_{0,0} + y_{2,0},$$
$$N_4 = y_{1,0} - x_{1,0},$$
$$N_5 = y_{1,0} - x_{2,0},$$
$$N_6 = y_{1,2} + y_{2,1},$$
$$N_7 = x_{2,0} - y_{2,0},$$

$$Q_1 = M_{10} + M_{11},$$
$$Q_2 = M_{10} - M_{14},$$
$$Q_3 = M_{17} - M_{18},$$
$$Q_4 = M_{19} - M_{22},$$
$$Q_5 = M_{11} + M_{16},$$
$$Q_6 = M_{15} + M_{17},$$
$$Q_7 = M_{20} + M_{22}.$$

Then

$$z_{0,0} = M_5 + Q_1 + M_{12},$$

$$z_{0,1} = M_8 + Q_2 + Q_3 + Q_4,$$

$$z_{0,2} = M_1 - Q_5 - M_{12} + Q_3 + Q_4,$$

$$z_{1,0} = M_6 - Q_1 + M_{13},$$

$$z_{1,1} = M_2 - Q_2 + M_{13} + Q_6,$$

$$z_{1,2} = M_9 - Q_5 + Q_6,$$

$$z_{2,0} = M_7 - Q_1 + Q_7 - M_{21},$$

$$z_{2,1} = M_3 - Q_2 - Q_3 + Q_7,$$

$$z_{2,2} = M_4 + Q_5 - Q_3 + M_{21}.$$

With a simple calculation we can verify that the above variant of Algorithm requires 38 additions less than the version of Makarov.

3. Principles of processing unit organization for 3×3 matrix product computing

The graph-structural model for realization of proposed algorithm is illustrated in Fig. 1. Boxes that are shared in gray indicate multipliers. Straight lines correspond to data transfer links. The rectangles marked with numbers 1 and 4 denote blocks of data permutation. Polygons with inscribed numbers 2, 3, 5 - denote blocks that contain sets of two-input, three-input, four-input and five-input adders. In particular, Figs. 2 and 5 show the patterns of data re-ordering in blocks 1 and 4, respectively. Fig. 3 shows the structure of block 2 which implements one half of the algebraic pre-additions, and Fig. 4 shows the structure of block 3, which implements the other half of the algebraic pre-additions. Fig. 6 presents the structure of block 5 which implements post-additions. Rectangles inscribed with signs of additions and subtractions are marked blocks of algebraic addition, as a rule of their operation is clear from context.

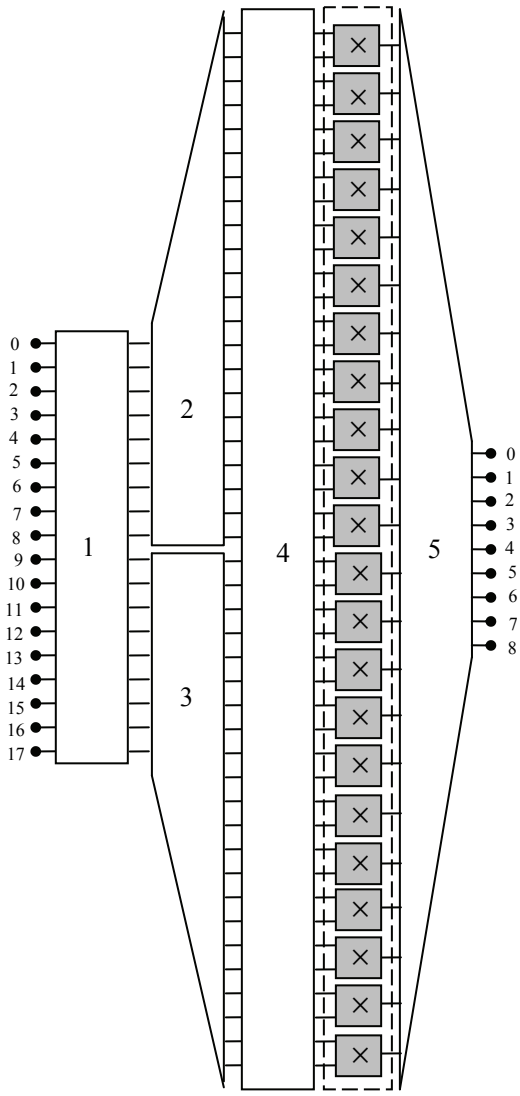


Fig. 1. Structural organization of a processing unit for 3×3 matrix product calculating corresponding to the proposed method
Rys. 1. Strukturalna organizacja jednostki procesorowej do wyznaczania iloczynu macierzy trzeciego stopnia zgodnie z proponowaną metodą

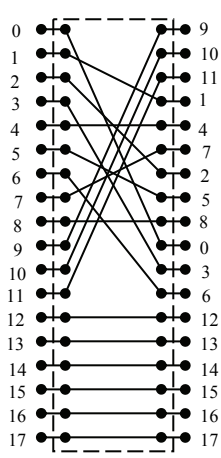


Fig. 2. Graphical representation of the principle of commutation and data transfer in block 1
Rys. 2. Prezentacja graficzna zasady komutacji oraz transferu danych realizowaną za pomocą bloku 1

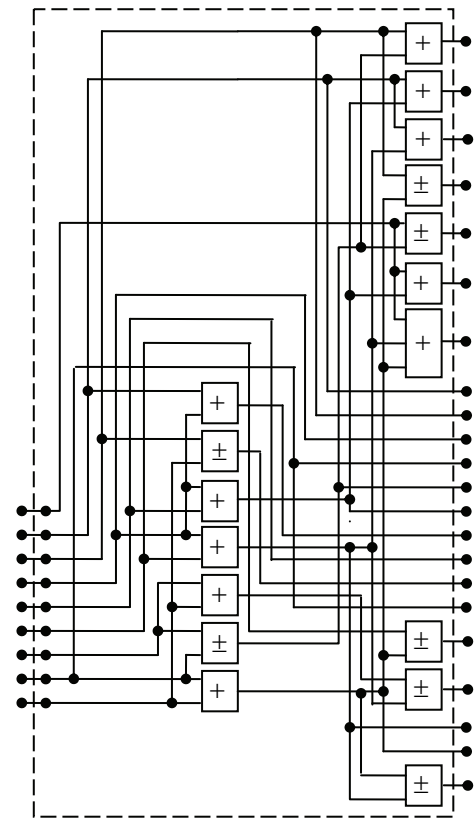


Fig. 3. The structure of processing unit 2 for calculating the top part of pre-additions
Rys. 3. Struktura bloku obliczeniowego 2 do realizacji górnej części przedsumowań

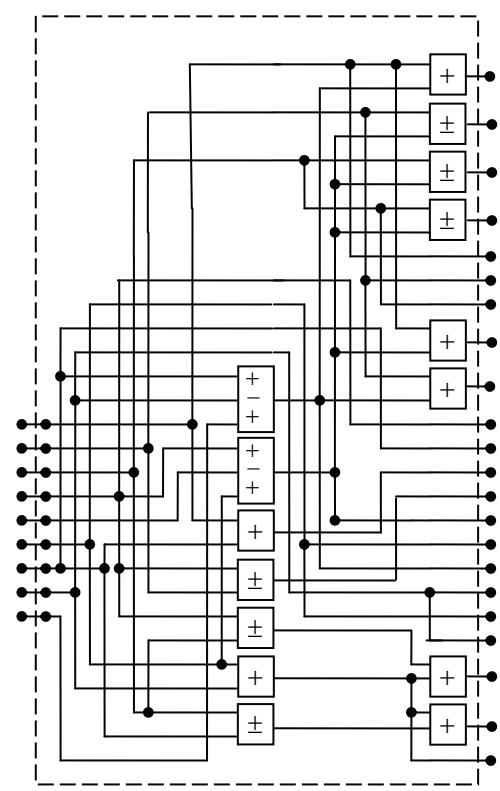


Fig. 4. The structure of processing unit 3 for calculating the down part of pre-additions
Rys. 4. Struktura bloku obliczeniowego 3 do realizacji dolnej części przedsumowań

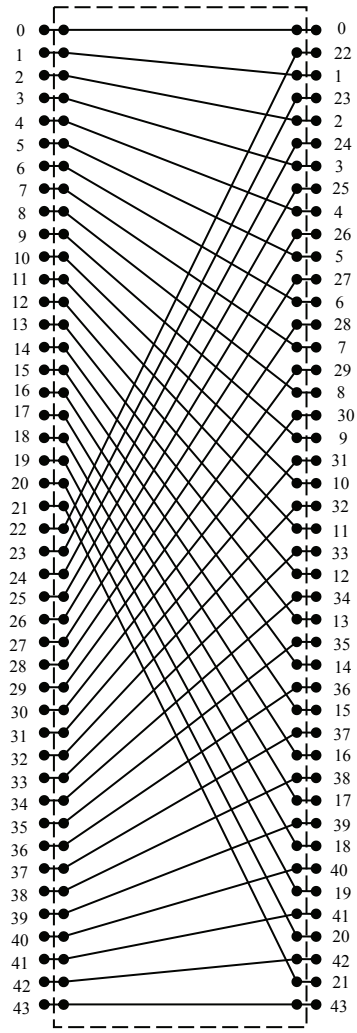


Fig. 5. Graphical representation of the principle of commutation and data transfer in block 4
Rys. 5. Prezentacja graficzna zasady komutacji oraz transferu danych realizowaną za pomocą bloku 4

4. Concluding remarks

The paper presents a new computational unit structure for implementing the 3×3 matrix multiplication. The principle of structure synthesis uses the same idea as the Makarov's method, but requires fewer summations (adders) and provides vectorization of computations of the matrix product. This allows the effective use of parallelization computational process of matrix multiplication and results in a reduction of the computation time.

Computer simulations prove that in the case of ASIC the complexity of the proposed structure implementation is smaller than that of the schoolbook (naïve) method completely parallel implementation for the number of bits of the multiplied operands greater than eight ($n > 8$). The implementation of this structure on the base of FPGA circuits that have built-in binary multipliers, also allows saving the number of blocks or realizing the whole matrix-multiplying unit with the use of a smaller number of simpler and cheaper FGPA circuits. For instance, a completely parallel implementation of a schoolbook (naïve) method on FGPA Xilinx Spartan-3 XC3S200 requires three such circuits, while the implementation of the proposed structures requires only two of them.

The work was conducted as a part of the cooperation agreement with the Institute of Information Technologies, Bulgarian Academy of Sciences.

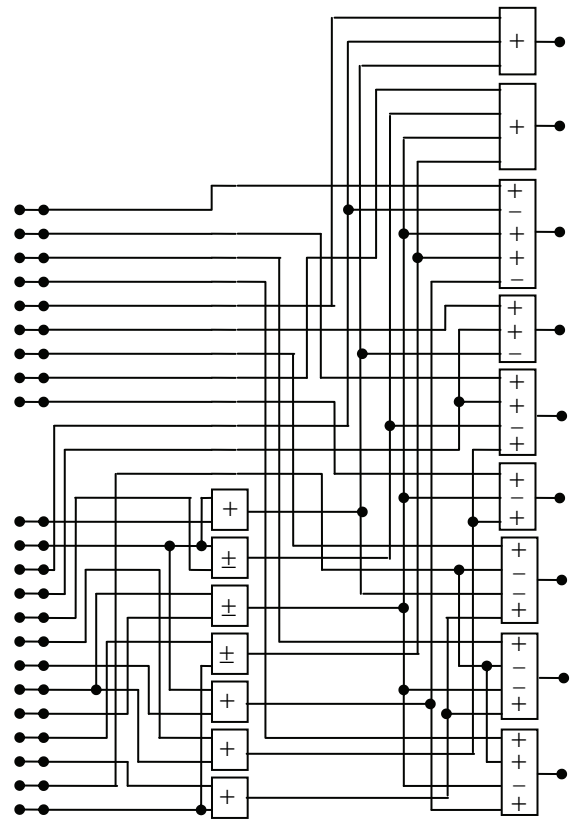


Fig. 6. The structure of processing unit 5 for realization of post-additions
Rys. 6. Struktura bloku obliczeniowego 5 do realizacji post sumowań

5. References

[1] Amira Bouridane A. and Milligan P.: Accelerating matrix product on reconfigurable hardware for signal processing, in Proc. 11th Int. Conf. Field-Programmable Logic Appl. (FPL), 2001, pp. 101–111.
[2] Jang J., Choi S. and Prasanna V. K.: Energy-efficient matrix multiplication on FPGAs, in Proc. Int. Conf. Field Programmable Logic Appl., 2002, pp. 534–544.
[3] Jang J. W., Choi S. and Prasanna V. K.: Area and time efficient implementations of matrix multiplication on FPGAs, in Proc. IEEE Int.Conf. Field Programmable Technol., 2002, pp. 93–100.
[4] Ţariov A.: Algorithmic models and structures of high-performance DSP processors, Szczecin, Informa, 2001. (in Russian).
[5] Charles-Éric Drevet, Md. Nazrul Islam and Éric Schost: Optimization techniques for small matrix multiplication, Theoretical Computer Science, Volume 412, Issue 22, 2011, pp. 2219-2236.
[6] Makarov M.: An algorithm for multiplying 3 x 3 matrics, Journal, USSR Computational Mathematics and Mathematical Physics archive Volume 26, Issue 1, May 1, 1987 pp. 293-294.
[7] Johnson R. and McLoughlin A.: Noncommutative bilinear algorithms for 3x3 matrix multiplication. SIAM J. Comput., 15(2): 595-603, 1986.
[8] Laderman J. D.: A noncommutative algorithm for multiplying 3 × 3 matrices using 23 multiplications, Bull. Amer. Math. Soc. 82 (1976), no. 1, 126–128.
[9] Nicolas T. Courtois, Gregory V. Bard and Daniel Hulme: A New General-Purpose Method to Multiply 3x3 Matrices Using Only 23 Multiplications., Miscellaneous papers in Computer and Information Science, Numerische Mathematik, 14 Aug. 20011: http://arxiv.org/abs/1108.2830.
[10] Hopcroft J. E. and Kerr L. R.: On minimizing the number of multiplications necessary for matrix multiplications, SIAM J. Appl. Math., 1971, vol 20, no.1, pp. 35-36.

otrzymano / received: 17.04.2012
przyjęto do druku / accepted: 01.06.2012 *artykuł recenzowany / revised paper*