Computational Intelligence and Information Technologies
June 20-21, 2001, Niš, Yugoslavia, 155-160

Synthesis of Folded FIR Filter Architecture With Reordered Partial Products

Ivan Milentijević, Teufik Tokić, Igor Nikolić, Oliver Vojinović and Ćirić Vladimir
Faculty of Electronic Engineering, University of Niš, Serbia
milentijevic@elfak.ni.ac.yu

Abstract

This paper describes a solution for folding of bit-plane FIR filter architecture. A transformation on DFG (Data Flow Graph) of source architecture, that enables the applying of folding technique, is proposed. The synthesized folded FIR architecture with reordered partial products has the reduced hardware-size at cost of time. The application of folding technique, where the folding factor is equal to the coefficient lenght, allows the bit-plane architecture to match the throughput requirement with minimized silicon area.

1. Introduction

FIR filtering is widely used in signal processing. Especially in the field of image processing, this algorithm becomes extremely demanding in term of computational throughput [1,2,3]. Regular structure of FIR filtering algorithm enables implementations based on systolic arrays. These array processors generally consist of a regular array of simple and nearly identical processing elements (PEs) in which data are communicated locally and operated on rhythmically [2,4,5]. Bit-level systolic arrays have shown to be an efficient approach for implementations in the area of high performance signal processing [1,6]. It is common to pipeline multipliers and to combine multiplications and the accumulation of products into an array. Sorting the accumulation of the partial products from the coefficient LSBs (Least Significant Bits) to the MSBs (Most Significant Bits) leads to the bit-plane architecture (BPA). The BPA is highly regular architecture, which allows extensive pipelining, regular layout, high computational throughput, truncation of LSBs of intermediate results without any loss of accuracy, and programmability of coefficients [1,6]. All these features point why the BPA is our candidate for folding.

In synthesizing DSP architectures, it is important to minimize the silicon area of the integrated circuits, which is achieved by reducing the number of functional units (such as multipliers and adders), registers, multiplexers, and interconnection wires. The folding transformation is used to systematically determine the control circuits in DSP architectures where multiple algorithm operations are time multiplexed to a single functional unit [7,8]. By executing multiple algorithm operations on a single functional unit, the number of functional units in the implementation is reduced, resulting in integrated circuit with low silicon area [8].

The aim of this paper is the synthesis of folded FIR filter architecture based on the BPA. However, the folding transformation can not be applied in a straightforward manner, because the algorithm is based on resorting of partial products. Multiplication can not be represented as node in the data flow graph (DFG). Here, we propose the transformation of source DFG, based on retiming and reordering of partial products, in order to enable the application of folding transformation. After that, we describe the application of folding technique and present the obtained folded architecture.

2. Fir filtering on bit plane semi-systolic architecture

Output words $\{y_i\}$ FIR filter are computed as

$$y_i = c_0x_i + c_1x_{i-1} + \ldots + c_{k-1}x_{i-k+1}, \quad (1)$$

where $c_0, c_1, \ldots, c_{k-1}$ are coefficients while $\{x_i\}$ are input words.

Computation (1) can be realized in different manners. When high performances are required systolic arrays are frequently used. Semi-systolic array share with systolic arrays desirable simplicity and regularity properties, in addition to their pipelining and multiprocessing schemes of operation. The only difference is that the broadcasting of data to many PEs in one time step is allowed in semi-systolic arrays, while systolic arrays are restricted to temporal locality of communication. Also, the existance of some additional connections can be allowed for semi-systolic architectures [1,5,6].

The bit–plane architecture (BPA) is semi-systolic architecture which provides regular connections with extensive pipelining and high computational throughput. The BPA is basic architecture for synthesis of folded architecture (FA), so we give a brief description of the BPA. In order to explain the BPA following notation is adopted:

m – coefficient word length,

k – number of coefficients $(c_0, c_1, \ldots, c_{k-1})$,

c_i^j – bit of coefficient c_i (with weight 2^j, and

n – input word length.

The BPA is obtained by resorting of the partial products of different multipliers as it is shown in Fig.1. In the first bit–plane the least significant partial products of all coefficients are computed and accumulated. The output of the first bit–plane is shifted by one weight and then the second lowest significant partial products are processed in the second bit plane and so on [1,6]. Starting bit-plane processing with the LSB's first, enables to truncate one LSB of the intermediate output signal after each bit–plane without any loss of accuracy in the more significant weights. This architecture is the basis for synthesis of the folded architecture (FA). The BPA with $k=3$, $m=4$ and $n=5$ is shown in Fig. 2.

3. Folding technique

The folding technique is introducted and detailly described in [7, 8]. With aim to clarify the applying of folding technique to the BPA we give a brief review of folding transformation [8].

The synthesis of folded data path is explained in Fig. 3 a) and Fig. 3 b). Fig 3 a) shows an edge $U \rightarrow V$ with $w(e)$ delays, while Fig. 3 b) depicts the corresponding folded data path. The data begin at the functional unit H_u which has P_u pipelining stages, pass through

$$D_F(U \rightarrow V) = Nw(e) - Pu + v - u \quad (2)$$

delays, and are switched into the functional unit H_v at the time instances $Nl+v$, where N is the number of operations folded to a single functional unit (folding factor), while u and v are the folding orders of nodes U and V that satisfy $N-1 \geq u$, $v \geq 0$ [8]. A folding set, S, is defined as an ordered set of operations, which contains N entries, executed by the same functional unit. For a folded system to be realizable, $D_F(U \rightarrow V) \geq 0$ must hold for all of the edges in the DFG. Once valid folding sets have been assigned, retiming can be used to satisfy this property or determine that the folding sets are not feasible.

After this short description of folding technique and involving of suitable notation, let us discuss a possible application of folding transformation to the BPA.

4. Synthesis of folded architecture

The folding transformation can not be applied to the BPA directly (Fig.1.), because the algorithm is based on resorting of partial products, so that multiplications of coefficients and input words are not recognized as operations, i.e. nodes in DFG (Fig. 1.). Each multiplication node in the DFG, shown in Fig. 1., represents one raw of basic cells (full adder and *and* gate) in the array described in Fig. 2. Thus, one multiplication is distributed through the whole array. Also, additions are performed in rows of basic cells and accumulation is provided by carry–save arithmetic along the array (Fig. 2.).

Our idea is to declare all operations in one plane to one operation "plane" and to provide time multiplexing to a single functional unit. In order to do that, we have to transform DFG from Fig. 1. Transformed DFG is shown Fig.4. Delays between planes are removed, as well as delays in the addition path. It allows simultaneous operation of plane units, but requires additional latches inside the plane. Transformed DFG is not suitable for implementation because of broadcasting line for input data words, but it is well prepared for folding.

Planes in the transformed DFG are denoted with dashed lines (Fig. 4). There is only one folding set $S=\{1,2,3,4\}$ which contains 4 operations "plane". The folding factor is equal to the coefficient length, i.e. $N=m=4$. There is internal pipelining, P_{int}, in each plane $P_{int} = k-1 =$

2 , but planes are connected via broadcast line for input words (without delays) and addition path (without delays, too). Thus, folding equations (2) for the determined folding set (Fig. 2.), where $w(e)=0$ and $P_v=0$, are

$$D_F(1\rightarrow 2) = 4\cdot 0 - 0 + 1 - 0 = 1$$
$$D_F(2\rightarrow 3) = 4\cdot 0 - 0 + 2 - 1 = 1$$
$$D_F(3\rightarrow 4) = 4\cdot 0 - 0 + 3 - 2 = 1.$$

The condition $D_F(U\rightarrow V) \geq 0$ is satisfied and additional retiming is not needed. Finally, transformed DFG from Fig.4. is mapped to the folded architecture (Fig. 5.).

Functional block diagram of folded architecture for $k=3$, $m=4$ and $n=5$ is shown in Fig. 6. Data flow for this example is given in Fig. 7., where X_A, X_B and X_C describe the presence of input data words at first, second and third row of cells, respectively, while C shows which coefficient bits are used. After the entering of new input word at first clock period all coefficient bits with weight 2^0 are available, zero values are passed through the entering multiplexers and sums of partial products are computed. At second clock period those sums are shifted for one weight right and reentered to the array for summation with partial products of weight 2^1. At the m-th clock period coefficient bits with weight 2^{m-1} are involved. Partial products with corresponding input words are formed and summations are performed. During the $(m+1)$-st clock period the result is obtained and new input value is entered. Each m clock periods one resulting y is generated (in Fig. 6. and Fig. 7. m=4). The initial latency for this architecture is m clock periods.

The corresponding transfer function is

$$G(z) = 2^{m-1}(c_0^{m-1}z^{-1} + c_1^{m-1}z^{-2} + \ldots + c_{k-1}^{m-1}z^{-k} + z^{-1/m}2^{m-2}(c_0^{m-2}z^{-1} + c_1^{m-2}z^{-2} + \ldots + c_{k-1}^{m-2}z^{-k} + \ldots + z^{-1/m}2^{0}(c_0^{0}z^{-1} + c_1^{0}z^{-2} + \ldots + c_{k-1}^{0}z^{-k}) \ldots) = (c_0 2^{m-1} + c_0 z^{-1/m}2^{m-2} + \ldots + c_0 z^{-(m-1)/m}2^{0})z^{-1} + (c_1 2^{m-1} + c_1 z^{-1/m}2^{m-2} + \ldots + c_1 z^{-(m-1)/m}2^{0})z^{-2} + \ldots + (c_{k-1}2^{m-1} + c_{k-1}z^{-1/m}2^{m-2} + \ldots + c_{k-1}z^{-(m-1)/m}2^{0})z^{-k} = z^{-1}(\Sigma_{i=0,k-1}(\Sigma_{j=0,m-1}c_i z^{-(m-j)/m}2^{j})z^{i}) = z^{-1}\Sigma_{i=0,k-1}c_i z^{i},$$

where $c_i z^{-(m-1-j)/m} = c_i^j$, z^{-1} stands for the delay equal to the period of signal ck1, and $2^{-1/m}$ denotes the delay equal to the clock period ck0.

5. Implementation

The synthesized folded architecture significantly reduces the hardware-size at cost of time, i.e. throughput, in respect to the source architecture. The unique multiplication and accumulation array is reduced with the factor m. Also, the total number of latches is decreased. Latches in carry and sum paths are removed, so the internal pipelining is not employed. Carries and sums are latched only for the folding at last row of the array. With aim to demonstrate the reduction in hardware size and its impact at time domain, we have implemented the folded architecture and the BPA in FPGA technology on chips XC4008E and XC4003E. Table I gives the survey of used hardware resources, while Table II shows time characteristics. Fig. 8. shows the implementation of the BPA and FA on chip XC4008E.

TABLE I The survey of used resources on FPGA chips

Arch.	chip	CLB	ff	LUT4	LUT3
BPA	XC4008	308(95%)	611	416	100
FA	XC4003	75 (75%)	113	128	30

TABLE II Time characteristics (T_0- clock period, f_0- clock frequency f- throughput)

Arch.	T_0[ns]	f_0[MHz]	f[MHz]
BPA	12.551	79.675	79.675
FA	23.101	43.476	8.645

6. Conclusion

The synthesis of folded FIR- filter architecture with reordered partial products has been presented. The application of folding technique has been made possible by the transformation of DFG of source architecture to the retimed DFG with broadcast line and reordered partial products. The transformed DFG has been found suitable for folding. The synthesized architecture has one functional unit (bit - plane) where all operations are multiplexed in time. The folding factor is equal to the coefficient length. For the sake of illustration of this solution, folded architecture is implemented in FPGA technology.

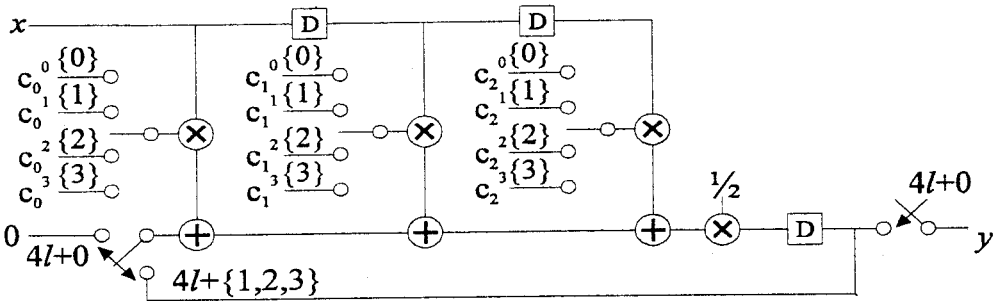


Fig.5. Folded architecture with folding set $S=\{1,2,3,4\}$.

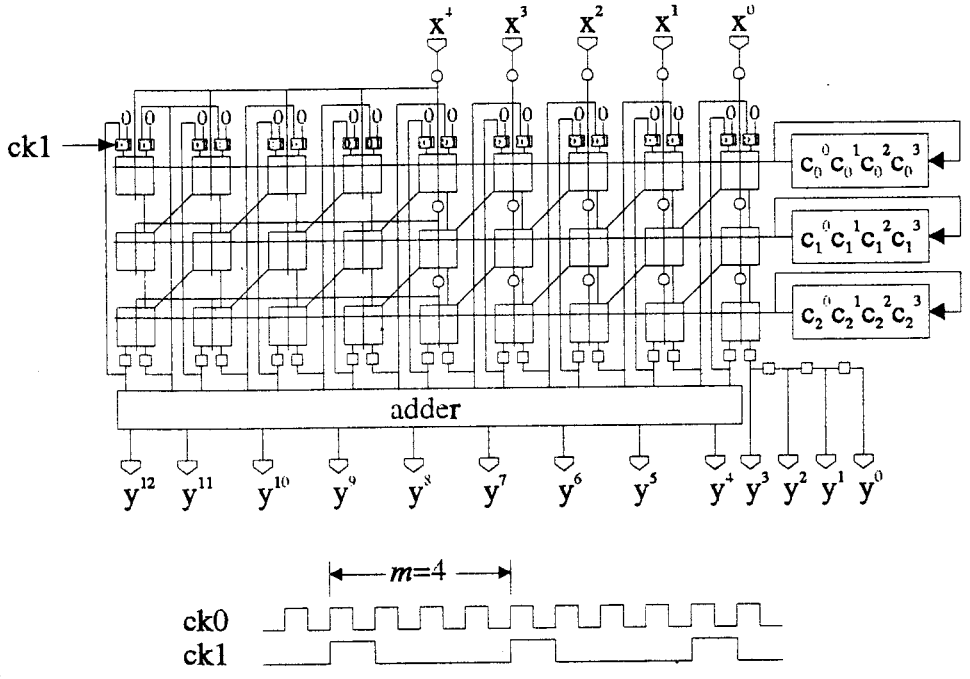


Fig. 6. Functional block diagram of the folded architecture for $k=3$, $m=4$ and $n=5$

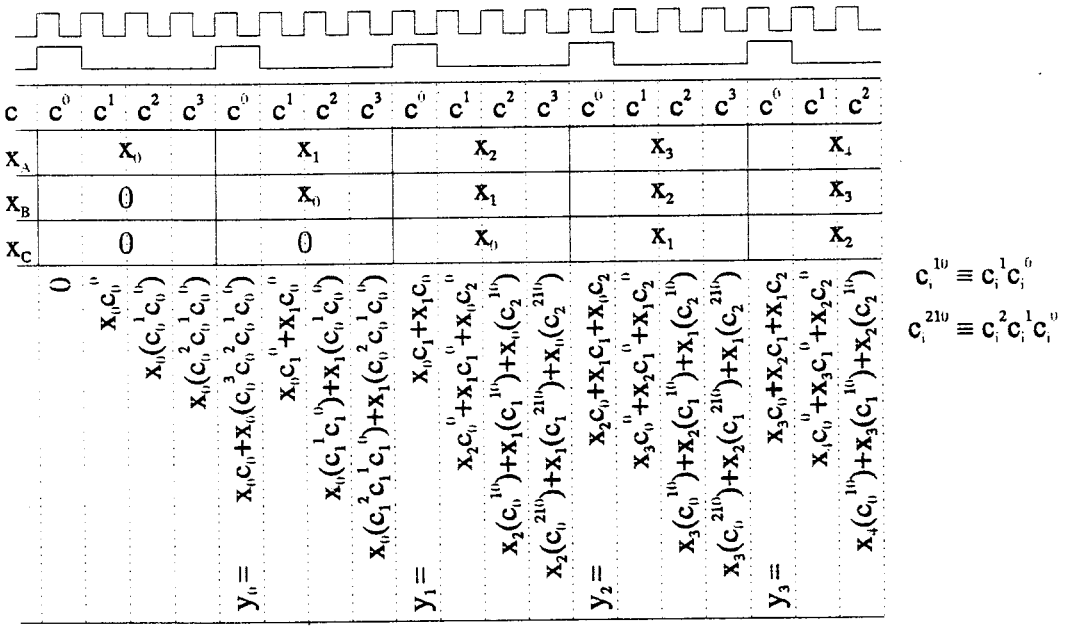


Fig. 7. Data flow for folded architecture ($k=3$, $m=4$ and $n=5$)

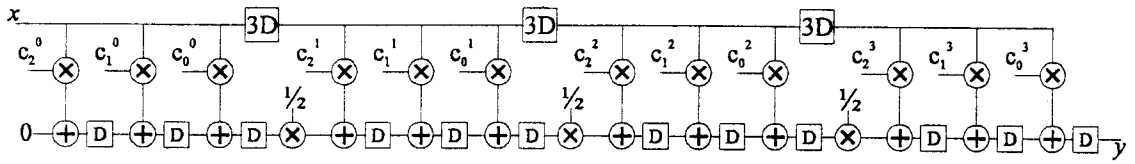


Fig. 1. The DFG (Data Flow Graph) for the BPA with $k=3$ and $m=4$

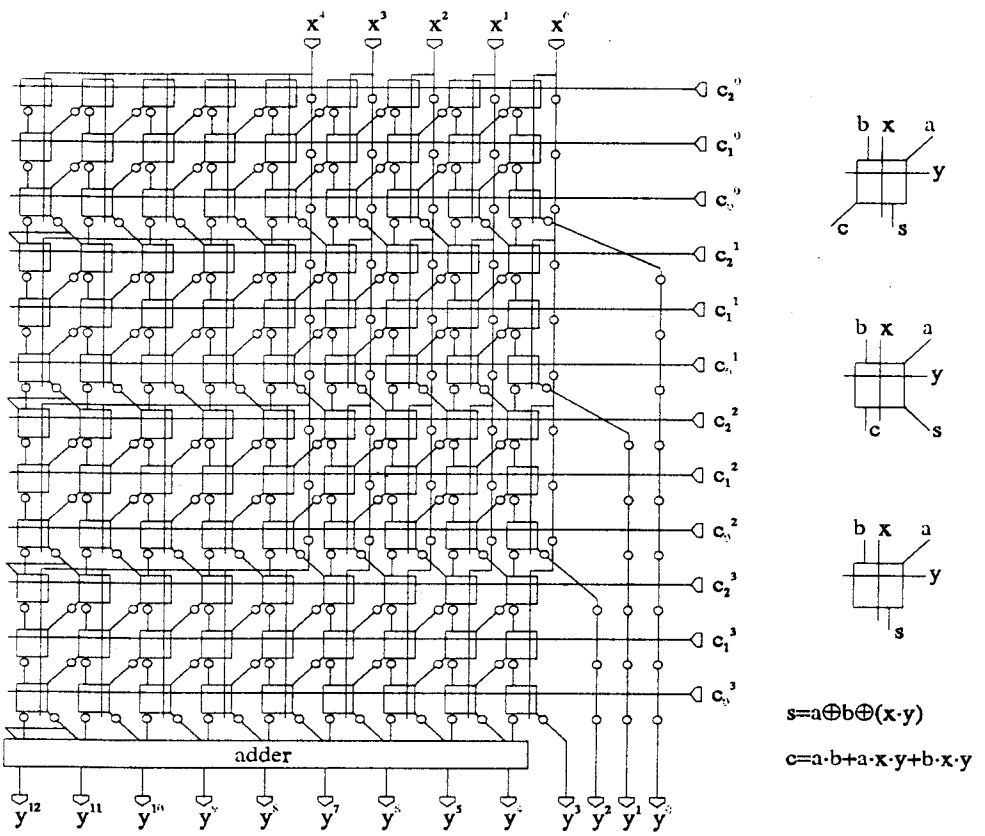
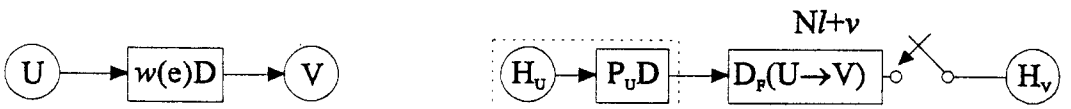


Fig. 2. The BPA for $k=3$, $m=4$ and $n=5$



a) An edge $U \rightarrow V$ with $w(e)$ delays b) The corresponding folded data path.

Fig. 3. The synthesis of folded data path

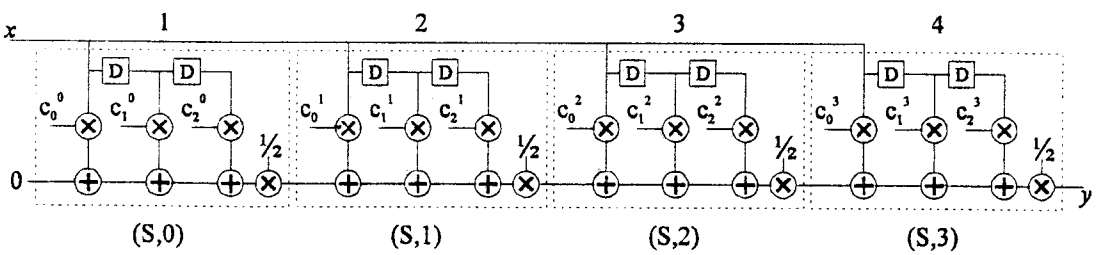


Fig. 4. Transformed DFG for $k=3$ and $m=4$

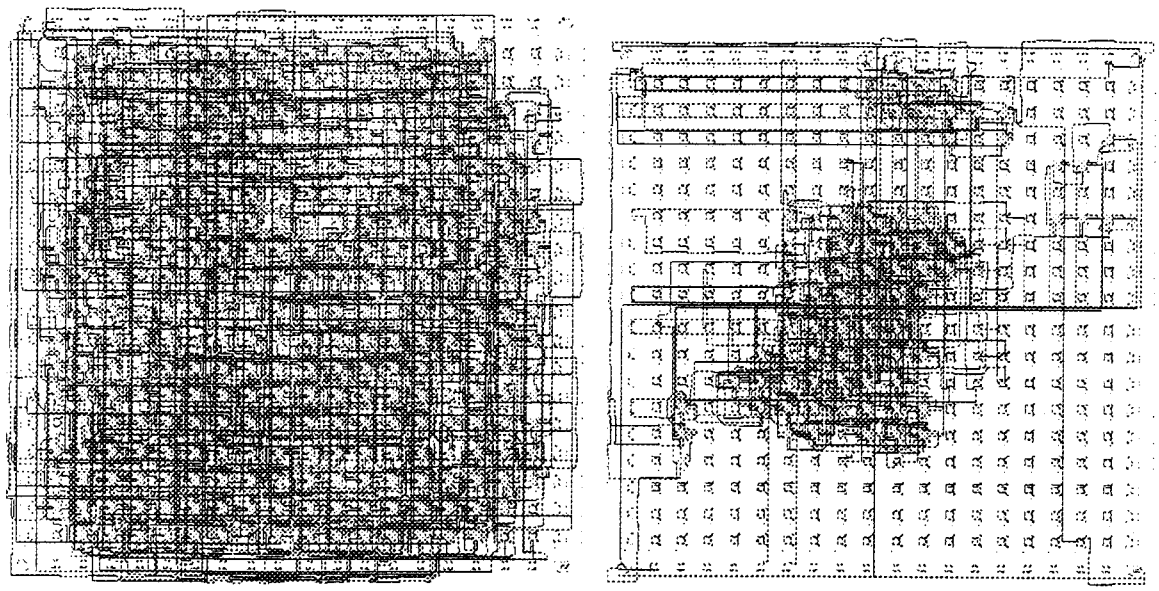


Fig. 8. The implementation of BPA and FA on chip XC4008E.

7. References

[1] T. Noll, "Semi-systolic Maximum Rate Transversal Filters with Programmable coefficients", *Workshop of Systolic Architectures*, Oxford, 1986, pp. 103-112.

[2] Y-C. Lin, F-C, Classes of Systolic Arrays for Digital Filtering, *Int. J. Electronics*, Vol. 70, No. 4, 1991, pp. 729-737.

[3] C-J. Chon, S. Mohanakrishnan, J. Evans, FPGA Implementation of Digital Filters, *ICSPAT '93*, 1943, pp1-9.

[4] I. Milentijević, I. Milovanović, E. Milovanović, M. Tošić, M. Stojčev, Two - Level Pipelined Systolic Arrays for Matrix - Vector Multiplication, Journal of Systems Architecture, *The EROMICRO Journal*, Vol. 44, No. 5, Feb. 1998, pp. 383 -387.

[5] I. Milentijević, M. S. Stojčev, D. Maksimović, Configurable Digit - Serial Convolver of Type F, *Microelectronics Journal*, Vol. 27. No. 6, Sep. 1996, pp. 559-566.

[6] D. Reuver, H. Klar, A configurable Convolution Chip with Programmable Coefficients, *IEEE Journal of Solid State Circuits*, Vol. 27, No. 7, July 1992, pp. 1121 - 1123.

[7] T. C. Denk, K. K. Parhi, Synthesis of Folded Pipelined Architectures for Multirate DSP Algorithms, *IEEE Transaction on Very Large Scale Integration (VLSI) Systems*, Vol. 6, No. 4, Dec. 1998, pp. 595-607.

[8] K. K. Parhi, *VLSI Digital Signal Processing Systems (Design and Implementation)*, John Wiley & Sons, In., New York, 2000.