Retiming and Register Number Minimization for Adaptive FIR Filter Architecture

V. Ciric, I. Milentijevic, O. Vojinovic
Faculty of Electronic Engineering, University of Nis, Serbia and Montenegro
{vciric, milentijevic, oliver}@elfak.ni.ac.yu

Abstract

Retiming of semi-systolic FIR filter architecture is proposed with aim to avoid usage of excessive amounts of registers. The focus is on preparation of architecture for folding and minimizing of number of registers that used in architecture. Set of techniques is applied to compute the minimum number of registers and to allocate the data to these registers. Using register minimization along with the folding transformation not only the number of functional units has been reduced but also the area consumed by memory in the folded architecture is kept to a minimum.

1. Introduction

The trends towards increasing data rates in digital signal processing systems have pushed the development and implementation of high-speed finite impulse response (FIR) digital filters beyond the capabilities of general purpose processors. Variety of approaches to custom implementation of FIR filters have been pursued. In order to attain high performance, parallel implementation strategies such as systolic methods, have been applied [1]-[2]. Thus, due to their geometrical regularity, they are suitable for VLSI implementations, either as stand-alone modules or as a part of complex digital data path. The choice of structure for the implementation of an FIR filter includes consideration of the factors such as hardware complexity and throughput. Many different structures exist, most of which provide some trade-off between complexity and throughput. Therefore, in order to establish optimal area-time tradeoff, a careful choice of circuit design style is necessary. 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. Folding provides a means for trading area for time in DSP architecture. 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 [3].

While the folding transformation reduces the number of functional units in the architecture, it may also lead to an architecture that uses a large number of registers. To avoid architectures using excessive amounts of registers, retiming is used to compute the minimum number of registers required to implement a folded DSP architecture. Allocation scheme in accordance to life time analysis is used to allocate data to these registers [3]. Using register minimization along with the folding transformation not only reduces the number of functional units but also keeps the area consumed by memory in the folded architecture to a minimum. Application of these techniques with aim to obtain a folded and retimed architecture for adaptive FIR filtering that has a minimum number of registers is the goal of this paper.

2. Background

The folding technique and retiming is introduced by K.K. Parhi and described in [3, 4]. With aim to clarify the register minimization for source architecture we give a brief review of retiming technique.

The data begin at the functional unit H_u that has P_u pipelining stages, pass through

$$D_F ?U ? \ V ?? Nw?e?? P_u ? v ? u \qquad (1)$$

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 $N ? 1 ? u, v ? 0$.

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, $D_F ?U ? \ V ?? 0$ must hold for all of the edges in the DFG. Once valid folding sets have been assigned, retiming can be used to either satisfy this property or determine that the folding sets are not feasible [3].

Using retiming, the number of delays on the edge $U\ ?\ \ V$ is changed from w(e) to

$$w_r(e)=w(e)+r(V)-r(U), \qquad (2)$$

where w_r(e) is the number of delays in edge $U\ ?\ \ V$ in the retimed DFG, and $r(X)$ denotes the retiming value for node X. Let $D_F'(U?\ \ V)$ denote the number of folded delays obtained by folding edge $U\ ?\ \ V$ in the retimed DFG. To ensure that the corresponding edge in the folded hardware has a nonnegative number of delays, the constraint $D_F'(U?\ \ V)?0$ must hold, which implies

$$Nw_r(e)\text{-}P_U+v\text{-}u?0. \qquad (3)$$

Using (1) and (2), an inequality (3) can be rewritten as

$$r(U)\text{-}r(V)\ ???D_F(U?\ \ V)/N?? \qquad (4)$$

If the solution for system of inequalities exists for adopted folding sets, the DFG can be retimed. If the retiming of DFG is possible, the life time analysis should be used to obtain minimal number of registers.

A data sample is live from the time it is produced through the time it is consumed. A variable occupies one register during each time that it is live. In life time analysis, the number of live variables at each time unit is computed and the maximum number of live variables at any time unit is determined. This is the minimum number of registers required for implementation [3].

Once the minimum number of registers has been determined, the data need to be allocated to these registers. Forward-backward register allocation is an allocation scheme that can be used to allocate data to the minimum number of registers [3]. Register allocation can be preformed using allocation table. The allocation scheme dictates how the variables are assigned to registers in the allocation table.

3. Basic architecture

In order to explain minimization of number of registers, the following notation is involved:

m_c – coefficient length
k_c – number of coefficients
c_i^j – bit of coefficient c_i (with weight 2^j)
N – folding factor
k – number of folding sets
L – total number of "operations" in the DFG, where one operation assumes forming of partial product and the addition performed on one "row" of basic cells (basic cell contains AND gate and full adder)
p – position of operation within the DFG ($0?p?L\text{-}1$).

DFG of the source architecture is shown in Fig. 1. The assignment of folding sets ($S_S|r$) onto the DFG from Fig. 1 that enables the synthesis of the folded FIR filter architecture with changeable both coefficient number and coefficient length is given in [9] and described with

$$s=p \bmod k$$
$$r=p \bmod N. \qquad (5)$$

Each Processing Element (PE) denoted using dashed boxes in Fig. 1 and marked as $p=0,1,\ldots,L\text{-}1$ belongs to one folding set described with (5).

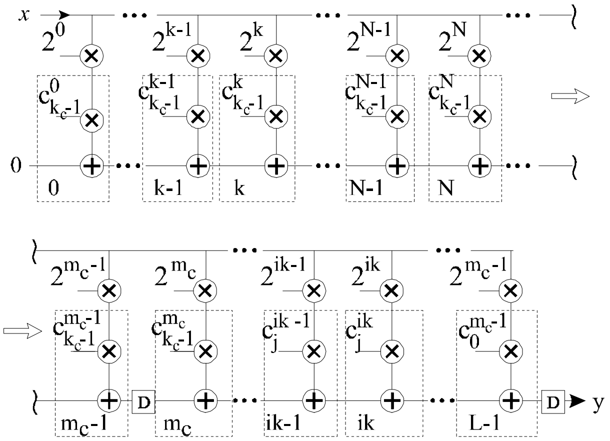

Fig 1. DFG of basic architecture for synthesis process

In order to have the answer to the question whether the system is foldable or not, we have to calculate the folding equations (1) and check the condition $D_f\,(V?\ \ V)\ ?\ 0$.

The general form of folding equation (1) related on two neighbor nodes U and V on positions p and $p+1$ from DFG shown in Fig. 1, with folding sets described with (5) is:

$$D_f\,(p?\ \ p+1)=N?w(e)\text{-}1+[(p+1) \bmod N]\text{-}[p \bmod N] =$$

$$= \begin{cases} ??\,?N\ ?\ 1?, & p \bmod ?N\text{ - }1??\ 0 \\ ?N\ ?\ 1, & p \bmod ?m_c\ ?\ 1??\ 0\ , \quad 0\ ?\ p\ ?\ L\text{-}2. \\ ?0, & otherwise \end{cases} \qquad (6)$$

4. Retiming

From (6), it can be seen that the condition $D_f\ (U?\ V)?0$ is not satisfied for nodes U and V on positions where $p_U \bmod (N\text{-}1) = 0$ and $p_V \bmod N = 0$, respectively. This is the main reason for retiming of the DFG from Fig. 1. Using the system of inequalities (4) and system of folding equations (6) the following system of inequalities is obtained

$$r(p)\text{-}\ r(p+1) = \begin{cases} ??\,1, & p \bmod ?N\text{ - }1??\ 0 \\ ?1, & p \bmod ?m_c\ ?\ 1??\ 0\ , \quad 0\ ?\ p\ ?\ L\text{-}2. \\ ?0, & otherwise \end{cases} \qquad (7)$$

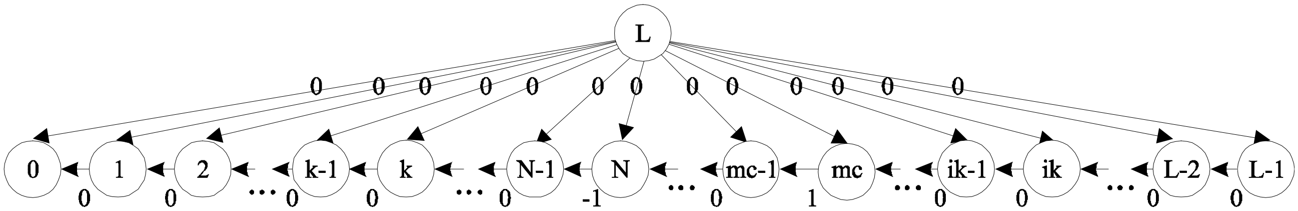

Fig. 2. Constraint graph

The constraint graph (Fig. 2) is formed with aim to provide the solution for inequalities (7). The constraint graph is directed graph where for each *r(p),p=1,2,…,L-1*, from (7) one node is assigned. Each inequality of type *r(U)-r(V)??* is represented by the directed edge from node *V* to node *U* with assigned weight *?*. An additional node denoted with *L* is connected with all other nodes with zero-weighted edges. The sum of weights on the shortest path, from node *L* to the node that corresponds to *r(p)*, represents the retiming for *r(p)*.

Now, when the constraint graph is derived let us highlight its important features. First feature relates on direction of edges (excluding edges that connect node L with other nodes). The edge is always directed from the node with higher position to the node with lower position, and connects only neighboring nodes. Second feature concerns edge index. The edge connecting nodes *U* and *V* on positions *i-1* and *i*, respectively, has edge index *i*. Thus, the weight of the edge with index *i* can be calculated as follows

$$w_i ? \begin{cases} 1, & i ? ?\frac{L ? n}{m_c}? \\ ?1, & i ? ?\frac{L ? n}{N}? , \; n=1,2, ... , L-1. \\ 0, & otherwise \end{cases} \qquad (8)$$

The number of edges with weight w_i*=-1*, as well as with weight w_i*=1* can be derived using previously described properties. The number of edges with weights w_i*=-1* according to (8), counting down from the node *L-1* to *p* is $?\frac{L ? p}{N}?$ times w_i and is equal to - $?\frac{L ? p}{N}?$.

The number of edges with weights w_i*=1* is $?\frac{L ? p}{m_c}?$.

Thus, the general form for retiming is:

$$r(p) ? ?\frac{L ? p}{m_c}? ? ?\frac{L ? p}{N}? \qquad (9)$$

In order to provide *r(p)?0*, we involve the following constraint m_c*? N*. The number of operations *L can be obtained as L=*k_c*?*m_c*=k?N*, which implies that number of coefficients (k_c) is less then the number of rows in folded architecture (*k*), i.e. $k_c<k$.

5. Minimization of Number of Registers

Bearing in mind that m_c*?1* and *N>1*, the lower and upper bounds for the retiming (9) are $r(0) ? ?\frac{L}{m_c}? ? ?\frac{L}{N}? ? k_c ? k$ and $r(L ? 1) ? ?\frac{1}{m_c}? ? ?\frac{1}{N}? ? 0$, respectively. The graphical representation of retiming is sketched in Fig. 3.

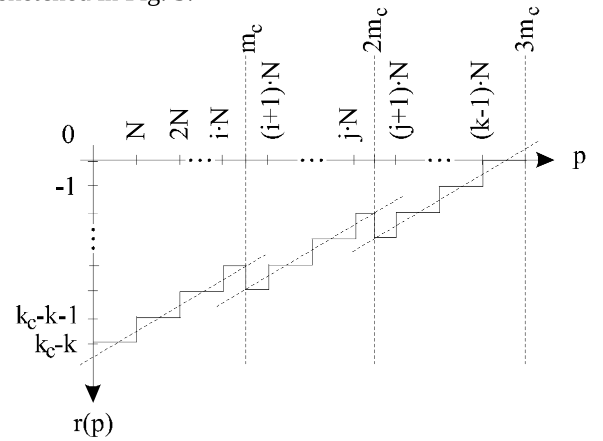

Fig. 3. Graphical representations of retiming for case *k*$_c$*=1, m*$_c$*=L*

Since $k_c<k$ all retiming values are placed onto negative part of *r(p)* axis. Fig. 3 and retiming (9) shows that the number of required successive input words *{x}*, used for simultaneous processing in architecture from Fig. 1, is $k ? k_c ? 1$. Actually, input word, x_i, is exploited as an operand in $k ? k_c ? 1$ successive computations on different nodes. For the chosen folding factor *N* (*N?1*) input data word x_i is active in the folded system for $T_{x_i} ? ?k ? k_c ? 1??N$ clock cycles.

The minimal number of registers, *R*, corresponds to the maximal number of active variables that is obtained for $k_c ? 1$. Thus, the minimal number of registers $R ? k ? 1 ? 1 ? k$. Since the folding factor is *N*, a new input data word will be entered into the folded system every *N* clock cycles. Graphical representations from Fig. 3 show that entering is required at $n ? m_c$ (*n=1,2,...,k*$_c$*-1*) time instances, also. According to the previous discussion graphical representation of parameterized data life cycle is generated (Fig. 4). The next step is the forming of allocation table from life time analysis in accordance to forward-backward allocation scheme (Fig. 5).

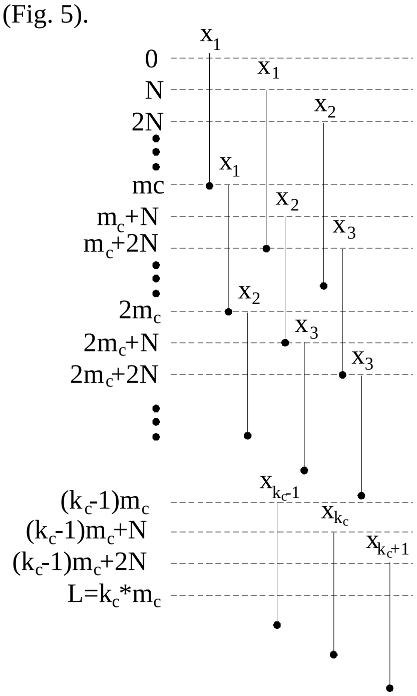

Fig. 4. Life time chart

	input	D_0	D_1		D_{k-1}
0	x_0				
1	x_0	x_0			
2	x_0		x_0		
⋮				…	
k-1	x_0				x_0
k	x_0	x_0			
⋮				…	
N	x_1				
N+1	x_1	x_1			
N+2	x_1		x_1		
⋮					
iN	x_i				
⋮		x_i			
m_c	x_i				
m_c*+1*	x_i	x_i			
⋮				…	
(i+1)N	x_i				
⋮					
(k-1)N	$x_{(k_c-1)m_c/N}$				
(k-1)N+1	$x_{(k_c-1)m_c/N}$	$x_{(k_c-1)m_c/N}$			
(k-1)N+2	$x_{(k_c-1)m_c/N}$	$x_{(k_c-1)m_c/N}$			
⋮				…	
*k*N-1*	x_{k-1}				

Fig. 5. Allocation table

The allocation table in Fig. 5 is the basis for the design of input data reordering module, with minimum number of registers, that enters the data into folded semi-systolic array for adaptive FIR filtering.

6. Conclusion

The solution for retiming of semi-systolic FIR filter architecture that enables the application of folding technique is proposed in this paper. Application of set of techniques onto a source semi-systolic architecture with aim to obtain a folded and retimed architecture for adaptive FIR filtering is presented. Allocation scheme in accordance with life time analysis is used to allocate data to these registers. Using register minimization along with the folding transformation not only the number of functional units has been reduced but also the area consumed by memory in the folded architecture has been kept to a minimum. The basis for the design of input data reordering module, with minimum number of registers, that enters the data into folded semi-systolic array for adaptive FIR filtering is derived.

7. References

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

[2] I. Milentijevic, M. S. Stojcev, D. Maksimovic, "Configurable Digit - Serial Convolver of Type F", *Microelectronics Journal*, Vol. 27. No. 6, Sep. 1996, pp. 559-566.

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

[4] 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.

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

[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] I. Milentijevic, V. Ciric, O. Vojinovic, T. Tokic, "Folded Semi-Systolic FIR Filter Architecture With Changeable Folding Factor", *Neural, Parallel & Scientific Computations, Dynamic Publishers,* Atlanta, Vol. 10, No 2, 2002, pp. 235-247.

[8] I.Z. Milentijevic, V. Ciric, T. Tokic and O. Vojinovic, "Folded Bit-Plane FIR Filter Architecture with Changeable Folding Factor", *DSD 2002, EUROMICRO - Digital System Design*, Dortmund, Germany, September 2002. pp. 45-52.

[9] I. Milentijevic, V. Ciric, "Assignment of Folding Sets for Adaptive FIR Filtering on Folded Array", *WPS-DSD 2003, EUROMICRO*, Belek, Turkey, September 2003. pp. 21-22.