A HIGH-PERFORMANCE FIR FILTER ARCHITECTURE FOR FIXED AND RECONFIGURABLE APPLICATIONS

S.Susmitha [1] T.Tulasi Ram [2]
susmitha449@gmail.com[1] ramttr0031@gmail.com[2]

1M. Tech Student, Dept of ECE, Vizag Institute of Technology, Dakamarri, Visakhapatnam, India.
2Assistant Professor, Dept of ECE, Vizag Institute of Technology, Dakamarri, Visakhapatnam, India.

Abstract: The power consumption and speed are the two main challenging factors in Very Large Scale Integrated Circuit (VLSI) design techniques. The computation saving is one of the way to obtain the optimized power consumption and speed. The design of finite-impulse response (FIR) filter using transpose form structure is naturally pipelined and upholds multiple constant multiplication (MCM) technique. This MCM technique results in large computation saving. But, the transpose form configurations are not supporting the block processing. In the existing method, the possibility of realization of FIR filter in transpose form configuration to achieve efficient area and delay for large order FIR filters were explored. In the FIR filter structure the ripple carry adder is used to add the partial inner products. The ripple carry adder provides efficient area utilization but its operating speed is slow. In this proposed method, the carry look ahead adder is used to increase the speed and also to reduce the area and power consumption. The proposed structure significantly reduces the area delay product (ADP) and energy per sample (EPS) than the existing FIR structure.

Keywords: Transpose form, ADP, EPS, Ripple carry adder, FIR, Block processing.

I. Introduction

Finite impulse response (FIR) digital filters are extensively used due to their key role in various digital signal processing (DSP) applications. Along with the advancement in very large scale integration (VLSI) technology as the DSP has become increasingly popular over the years, the high speed realization of FIR filters with less power consumption has become much more demanding. Since the complexity of implementation grows with the filter order and the precision of computation, real-time realization of these filters with desired level of accuracy is a challenging task. Several attempts have, therefore, been made to develop dedicated and reconfigurable architectures for realization of FIR filters in application specific integrated circuits (ASIC) and field programmable gate arrays (FPGA) platforms. Systolic designs represent an attractive architectural paradigm for efficient hardware implementation of computation-intensive DSP applications, being supported by the features like simplicity, regularity and modularity of structure. Additionally, they also possess significant potential to yield high-throughput rate by exploiting high-level of concurrency using pipelining or parallel processing or both. To utilize the advantages of systolic processing, several algorithms and architectures have been suggested for systolization of FIR filters. However, the multipliers in these structures require a large portion of the chip-area, and consequently enforce limitation on the maximum possible number of processing elements (PEs) that can be accommodated and the highest order of the filter that can be realized. Multiplier less distributed arithmetic (DA)-based technique, has gained substantial popularity, in recent years, for their high-throughput processing capability, and increased regularity which results in cost-effective and area-time efficient computing structures. The main operations required for DA-based computation of inner-product are a sequence of look-up-table (LUT)-accesses followed by shift accumulation operations of the LUT output. DA-based computation is well-suited for FPGA realization, because the LUT as well as the shift-add operations can be efficiently mapped to the LUT-based FPGA logic structures.

In FIR filtering, one of the convolving sequences is derived from the input samples while the other sequence is derived from the fixed impulse response coefficients of the filter. This behavior of FIR filter makes it possible to use DA-based technique for memory-based realization. It yields faster output compared with the multiplier-accumulator-based designs because it stores the pre-computed partial results in the memory elements, which can be read out and accumulated to obtain the desired result. The memory requirement of DA-based implementation for FIR filters, however, increases exponentially with the filter order. DA was first introduced by Croisieret al; and further developed by Peled and Lui for efficient implementation of digital filters. Attempts are made to use offset-binary coding to reduce the ROM size by a factor of 2. An LUT-less adder-based DA approach has been suggested by Yoo and Anderson, where memory-space is reduced at the cost of additional adders. Memory-partitioning and multiple memory-bank approach along with flexible multi-bit data-access mechanisms are suggested for FIR filtering and inner-product computation in order to reduce the memory size of DA-based implementation. Allredet al have suggested an efficient DA-based implementation of least mean square (LMS) adaptive filter using a decomposition of DA based FIR computation and subsequent memory decomposition. All these structures, however, are not suitable for implementation of the FIR filters in systolic hardware since the partial products available from the partitioned memory modules are summed together by a network of output adders. A new tool for the automatic generation of highly parallelized FIR filters based on PARO design methodology. Where the authors have performed hierarchical partitioning in order to balance the amount of local memory with external communication, and they have achieved higher throughput and smaller latencies by partial localization. A systolic decomposition technique is suggested in a recent paper for memory-efficient DA-based implementation of linear and circular convolutions. In this paper we have extended further the work of to obtain an area-delay-power-efficient implementation of FIR filter in FPGA platform.

II. Literature Survey

Pramod Kumar Meher (2006) proposed the structure that involves significantly less memory and less areadelay complexity compared with the existing DA-based structures for circular convolution. Besides, it is shown that the proposed systolic designs for circular convolution can be used for computation of linear convolution as well.

Basant Kumar Mohanty and Pramod Kumar Meher (2015) explore the possibility of realization of block FIR filter in transpose form configuration for area-delay efficient realization of large order FIR filters for both fixed and reconfigurable applications.

Yu Pan and Pramod Kumar Meher(2014)proposed the resource minimization problem in the scheduling of adder-tree operations for the MCM block, and presented a mixed integer programming (MIP) based algorithm for more efficient MCM-based implementation of FIR filters. Experimental result shows that up to 15% reduction of area and 11.6% reduction of power (with an average of 8.46% and 5.96% respectively) can be achieved on the top of already optimized adder/subtractor network of the MCM block.

Abbes Amira, Pramod Kumar Meherand ShrutisagarChandrasekaran (2008) presented the design optimization of one and two dimensional fully pipelined computing structures for efficient implementation of finiteimpulse-response (FIR) filter to obtain effective area, delay and power by using systolic decomposition of innerproduct computation based on distributed arithmetic (DA). The systolic decomposition scheme is found to offer a flexible choice of the address length of the lookup tables (LUT) for DA-based computation to decide on suitable area time trade off. It is observed that by using smaller address lengths for DA-based computing units, it is possible to reduce the memory size, but on the other hand that leads to increase of adder complexity and the latency.

III. Existing System

Adaptive Algorithms

There are numerous methods for the performing weight update of an adaptive filter. There is the wiener filter, which is the optimum liner filter in terms of mean squared error, and several algorithms that attempt to approximate it, such as the method of steepest descent. There is also least-mean square algorithm, developed by Windrow and Hoff originally for use in artificial neural networks. Finally, there are other techniques such as the recursive-least square algorithm and the kalman filter. The choice of algorithm is highly dependent on the signals of interest and the operating environment, as well as the convergence time required and computation power available.

Problem Statement

Due to the high performance requirements and increasing complexity of

DSP and multimedia communication applications, filters with large number of taps are required to increase the performance in terms of high sampling rate. As a result the filtering operations are computationally intensive and more complex in terms of hardware requirements. The FIR filters perform the weighted summations of input sequences with constant coefficients in most of the signal processing and multimedia applications. These filters are widely used in video convolutions functions, signal preconditioning, and other communication applications. The decrease in computational complexity causes the increase in the performance, in terms of speed, area and power. High speed, low area and power efficient conscious design techniques in SoC include efforts at all level of abstraction. One way to efficiently incorporate high performance design technique is to implement IP cores.

These cores have following major advantages.

- Reusability across designs
- Reduction of the design effort
- Shorter time to market.

The disadvantage of FIR filters is that they require high order. The high order demands more hardware, area and power consumption. To minimize these parameters, our goal is to implement an efficient high order filter in digital systems. By the reduction of arithmetic in terms of multipliers, our goal is to reduce the parameters namely, hardware, area and power. This is ultimate goal of the implementation of an efficient FIR filter and hence DA algorithm is used for implementation of high order FIR filter. FIR filter is incorporated with a MAC unit. The purpose of MAC unit is to multiply the input with constant coefficients, to shift and then to add them. This process is repeated until all partial products produce the output after accumulation. It increases the hardware complexity because a simple multiplier circuitry is used. The idea is to somehow bypass or replace the multiply and shift operations with less complex operations. Distributed Arithmetic (DA) Algorithm can be used to replace MAC unit. The DA Algorithm actually uses lookup table for storing constant coefficients. So the use of lookup tables reduces the hardware complexity and hence the new design is more efficient in terms of less area, more speed and low power consumption. FIR filter reference core uses a simple MAC unit. We have replaced MAC unit in FIR filter reference core with DA Algorithm. In this study, performance of Reference Core with Simple MAC and reference core with DA is compared.

DFG Transformation

The computation of DFT-3 and DFT-4 can be realized by DFG-3 of non overlapping blocks, as shown in Fig. 1. We refer it to block transpose form type-I configuration of block FIR filter. The DFG-3 can be retimed to obtain the DFG-4 of Fig. 5, which is referred to block transpose form type-II configuration. Note that both type-I and type-II configurations involve the same number of multipliers and adders, but type-II configuration involves nearly L times less delay elements than those of type-I configuration. We have, therefore, used block transpose form type-II configuration to derive the proposed structure. We present mathematical formulation of block transpose form type-II FIR filter for a generalized formulation of the concept of block-based computation of transpose form FIR filers.

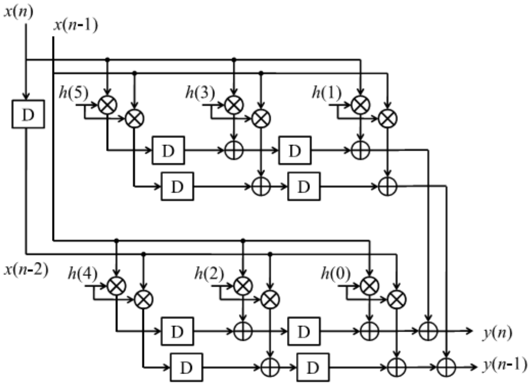


Fig. 1. Merged DFG (DFG-3: transpose form type-I configuration for block FIR structure).

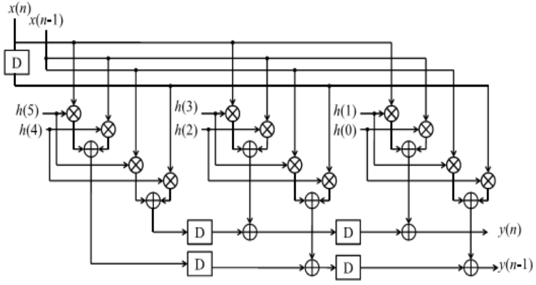


Fig. 2. DFG-4 (retimed DFG-3) transpose form type-II configuration for block FIR structure.

IV.Proposed System

There are several applications where the coefficients of FIR filters remain fixed, while in some other applications, like SDR channelizer that requires separate FIR filters of different specifications to extract one of the desired narrowband channels from the wideband RF front end. These FIR filters need to be implemented in a RFIR structure to support multi standard wireless communication. In this, we present a structure of block FIR filter for such reconfigurable applications. In this section, we discuss the implementation of block FIR filter for fixed filters as well using MCM scheme.

The proposed structure for block FIR filter is based on the recurrence relation of (12) for the block size L = 4. It consists of one coefficient selection unit (CSU), one register unit (RU), M number of inner product units (IPUs), and one pipeline adder unit (PAU). The CSU stores coefficients of all the filters to be used for the reconfigurable application. It is implemented using N ROM LUTs, such that filter coefficients of any particular channel filter are obtained in one clock cycle, where N is the filter length. The RU [shown in Fig. 4(a)] receives xk during the kth cycle and produces L rows of S0 k in parallel. L rows of S0 k are transmitted to M IPUs of the proposed structure. The M IPUs also receive M short-weight vectors from the CSU such that during the kth cycle, the (m + 1)th IPU receives the weight vector cM−m−1 from the CSU and L rows of S0 k form the RU. Each IPU performs matrix-vector product of S0 k with the short-weight vector cm, and computes a block of L partial filter outputs (rk m). Therefore, each IPU performs L inner-product computations of L rows of S0 k with a common weight vector cm. The structure of the (m +1)th IPU is shown in Fig. 4(b). It consists of L number of L-point inner-product cells (IPCs). The (l +1)th IPC receives the (l +1)th row of S0 k and the coefficient vector c m, and computes a partial result of inner product r(kL − l), for 0 ≤ l ≤ L − 1. Internal structure of (l + 1)th IPC for L = 4 is shown in Fig. 5(a). All the M IPUs work in parallel and produce M blocks of result (rk m). These partial inner products are added in the PAU [shown in Fig. 5(b)] to obtain a block of L filter outputs. In each cycle, the proposed structure receives a block of L inputs and produces a block of L filter outputs, where the duration of each cycle is T = TM + TA + TFA log2L, TM is one multiplier delay, TA is one adder delay, and TFA is one full-adder delay.

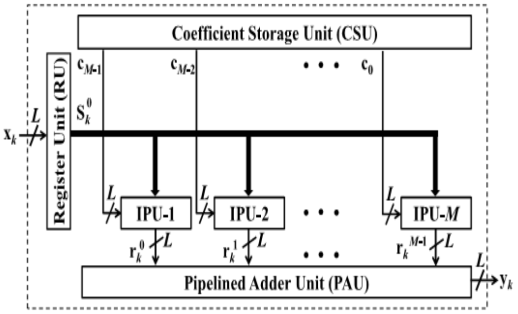


Fig. 3. Proposed structure for block FIR filter.

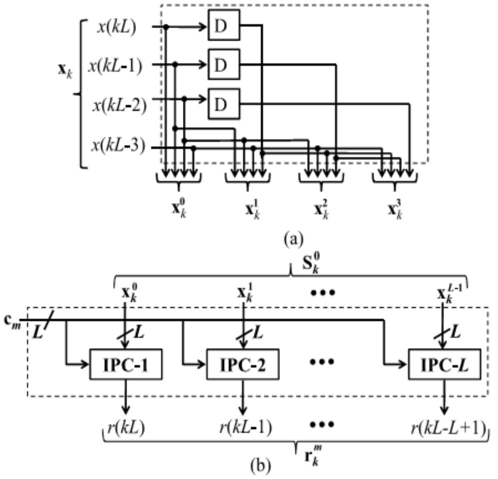


Fig. 4. (a) Internal structure of RU for block size *L* = 4. (b) Structure of (*m* + 1)th IPU.

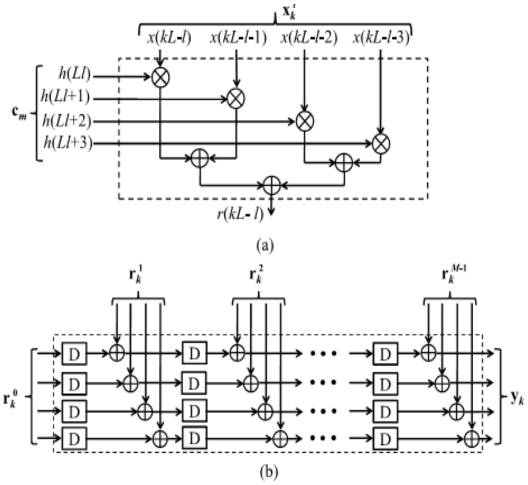
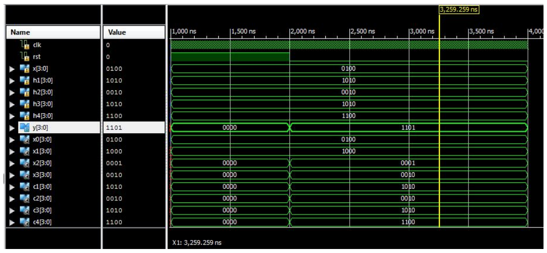


Fig. 5. (a) Internal structure of (*l* + 1)th IPC for *L* = 4. (b) Structure of PAU for block size *L* = 4.

V.RESULTS

Implementation of FIR filter cores has been observed and we can see that fir filter cores have been implemented with both reference and DA structure. Results have been taken in terms of area utilized, and speed performance. FIR filter cores have been designed in Verilog HDL and implemented using Xilinx 13.2 tool. Simulations were performed using iSim.

Simulation Result:

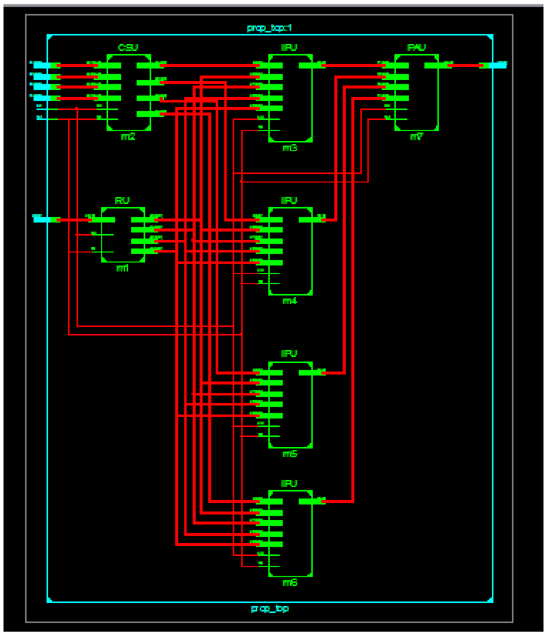


Name	Value
clk	0
rst	0
x[3:0]	0100
h1[3:0]	1010
h2[3:0]	0010
h3[3:0]	1010
h4[3:0]	1100
y[3:0]	1101
x0[3:0]	0100
x1[3:0]	1000
x2[3:0]	0001
x3[3:0]	0010
c1[3:0]	1010
c2[3:0]	0010
c3[3:0]	1010
c4[3:0]	1100

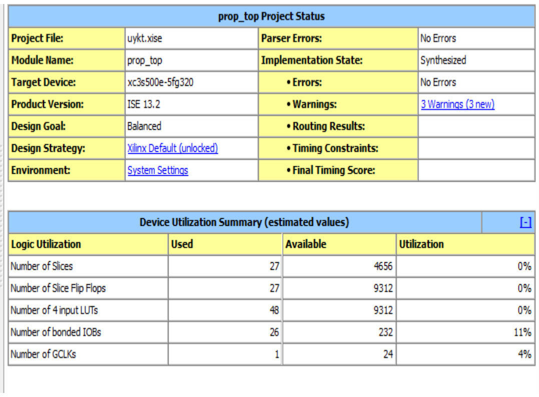
X1: 3,259.259 ns

Synthesis Results:

RTL Schematic:



Design Summary:



prop_top Project Status			
Project File:	uykt.xise	**Parser Errors:**	No Errors
Module Name:	prop_top	**Implementation State:**	Synthesized
Target Device:	xc3s500e-5fg320	**• Errors:**	No Errors
Product Version:	ISE 13.2	**• Warnings:**	3 Warnings (3 new)
Design Goal:	Balanced	**• Routing Results:**	
Design Strategy:	Xilinx Default (unlocked)	**• Timing Constraints:**	
Environment:	System Settings	**• Final Timing Score:**	

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	27	4656	0%
Number of Slice Flip Flops	27	9312	0%
Number of 4 input LUTs	48	9312	0%
Number of bonded IOBs	26	232	11%
Number of GCLKs	1	24	4%

Conclusion

In this paper, the possibility of realization of block FIR filters in transpose form configuration for area and delay efficient realization of fixed FIR applications were explored and also the impact of power consumption, delay, area has been analyzed. Simulation results have been calculated. The use of ripple carry adders in existing method increases area and power consumption. To overcome this drawback, carry look ahead adder is used in the proposed method. The proposed structure has the best results in the reduction of number of slices, LUTs, power consumption, area delay product, energy per sample than the existing method for higher order FIR filter.

REFERENCES

[1]. Farhat Abbas Shah, Habibullah Jamal, Muhammad Akhter Khan, "Reconfigurable Low Power FIR Filter based on Partitioned Multipliers", 16-19 December, KFUPM, Dhahran, KSA, ICM, 2006.

[2]. Xilinx, "Design Reuse Methodology for ASIC and FPGA Designers", 2004. www.xilinx.com.

[3]. N. Sankarayya, K.Roy, D. Bhattacharya, "Algorithms for Low Power and High Speed FIR filter Realization Using Differential Coefficients", Analog and Digital Signal Processing, IEEE Transactions, vol, 44(6), pp. 488-497, 1997.

[4]. A.T.Erdogan, M.Hasan and T.Arslan, "Algorithmic low power FIR cores", circuits and systems, IEEE proceedings, vol 150, pp. 155-169, 2003.

[5]. Heejongyoo and David V. Anderson, "Hardware efficient distributed arithmetic architecture for high order digital filters", Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE, vol 5, pp. 125-128, 2005.

[6]. Wang sen, Tang Bin and Zhu Jun, "Distributed arithmetic for FIR filter design on FPGA", Communications, Circuits and Systems, 2007. ICCCAS 2007. IEEE, pp. 620-623, 2007.

[7]. Patrick Longa and Ali Miri, "Area efficient FIR filter design on FPGAs using distributed arithmetic", Symposium on signal processing and information technology, IEEE processing, pp. 248-252, 2006.