FIR Filter Design for ASIC and FPGA Realization using DA-Approach

Rajupalem Adithya[1] Mr. K. Bala[2]
[1]M.Tech Student [2]Associate Professor
[1,2]Department of Electronics and Communication Engineering
[1,2]Srinivasa Institute of Technology and Science, Kadapa, Andhra Pradesh, India

Abstract— This paper presents approaches for high-throughput with Distributed Arithmetic (DA) based implementation of reconfigurable Finite Impulse Response (FIR) digital filters whose filter coefficients will be changing during runtime. Conventionally, for DA-based implementation of reconfigurable FIR digital filter, the lookup tables (LUTs) are needs to be implemented in RAM, but the RAM based LUT is found to be costly. Therefore, a shared-LUT design is proposed for the realization of DA computation. Instead of using separate registers for storing possible results of partial inner products for DA processing of different bit positions, registers are shared by the DA units for bit slices of different weightage. The proposed design is having nearly less area-delay product, when compared to DA-based conventional structure. The critical operations of the FIR filter are multiplication and accumulation. But Real-time signal processing requires a unit which consumes low power, and having high throughput and high speed Multiplier-Accumulator (FIR) unit, which is always important to achieve a high performance digital signal processing system. To achieve the parallel response of the FIR Filter instead of serial response for the given serial input, DA- Algorithm approach is introduced.

Key words: FIR Filter, Distributed Arithmetic (DA) Algorithm, Reconfigurable Implementation, Look-Up-Table (LUT)

I. INTRODUCTION

Digital signal processing (DSP) algorithm implementation on hardware requires efficient analysis of area and throughput with high performance and low cost. These algorithms on field programmable gate array (FPGA) and ASIC provide programmable and dedicated hardware design by enabling fast digital circuit prototypes. The FIR filters are linear filters that perform shifting, and multiplication operation for K filter taps and provides the output that depends on the present and past (k-1) input samples. The FIR filters require k multiply and accumulate (MAC) operations.

The MAC operation is replaced by look up tables (LUTs) and summations. In DA technique the LUT is used to store the pre-calculated sum of products and during each cycle the least significant bits of each tap is concatenated to form the address for the LUT. The DA technique provides high throughput since it generates the filter output in least number of clock cycles that depends on the input bit precision. The elimination of hardware multipliers reduces the logic complexity in DA. Due to advancement in memory design technology, memory size is reduced, which makes the DA worth considerable.

The adaptive filter is widely used in various applications such as channel equalization, system identification, interference cancellation, echo cancellation, radar and sonar signal processing. For time varying environment, the LMS algorithm is used as an adaptation rule which is commonly used in various hardware requirements. The different approaches have been made to implement the DA based adaptive filter, but these are undesirable for practical applications.

The high throughput has been achieved at the cost of the increased number of adder units, which makes inefficient use of chip area. The low-power dissipation has achieved at the cost of the increase in memory usage. DA based adaptive filter implementation poses many challenges such as updating the weight during each sampling period requires the update of entire LUT. It degrades the DA based filter performance with large filter size.

II. DA FIR FILTER

The K_{th} order discrete time FIR filter provides the output y(n) as a sum of scaled and delayed input samples x(n) as follows:

$$y(n) = \sum_{k=0}^{K-1} w_k x(n-k) \quad (1)$$

DA is a bit serial arrangement to calculate the inner product of the pair of vectors by storing all combinations of the sum of filter weights in LUT. The N- bit 2's complement binary number is represented as:

$$x(n-k) = -b_{k0} + \sum_{n=1}^{N-1} b_{kn} 2^{-n}, k = 0, 1, \cdots, K-(1) \quad (2)$$

Substituting (2) in (1) and swapping the order of summation yields

$$y(n) = -\sum_{k=0}^{K-1} w_k b_{k0} + \sum_{n=1}^{N-1} [\sum_{k=0}^{K-1} w_k b_{kn}] 2^{-n} \quad (3)$$

From (3) it is concluded that a binary AND operation are performed between a single bit of input variable and all the bits of the weights. The exponential factors indicate the scaling of each bit. The term in the square braces of (3) consider only one value among 2^k possible values stored in DA filter LUT (DA-F-LUT), and the output is generated after N clock cycles.

These values can be pre-calculated for all values of k which are used to fill the LUT table of 2^k words and each is addressed by k bits. DA FIR filter implementation for k= 4 is shown in Figure 1.The DA-F-LUT stores all the possible combination sums of filter weights. The concatenation of the rightmost bits of input taps becomes the address of DA-F-LUT. The shift register is used to shift the bits right by one place for each accumulated value. The DA-F-LUT entries are shifted and accumulated N times to generate the filter output. The sign control is used to change addition to subtraction for sign bits, which are included in the first expression of (3).

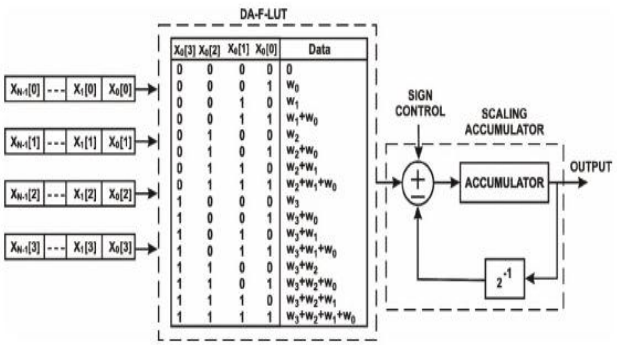

Fig. 1: Conceptual diagram of DA FIR filter for 4 tap.

III. PROPOSED FIR DIGITAL FILTER FOR ASIC IMPLEMENTATION USING DA-APPROACH

The proposed DA-based structure of Reconfigurable FIR Digital filter is shown in Fig. 2. The input bits Xin1, Xin2…… arriving at every sampling instant are fed into the Serial-In Parallel-Out shift register. The Serial-In Parallel-Out shift register will decompose the N samples to P vectors. P Vectors are fed into Reconfigurable Partial Product Generators (RPPGs). The RPPG will take 2 input bits and it will generate the partial products and corresponding bit slices in parallel using the LUT composed of a single register bank. RPPG consists of Registers and MUXes. The inputs to the MUX are Sample Input bits and the control word.

Storage consumption can be reduced by sharing each Look-Up-Table across the bit slices.

In order to access the contents of LUT simultaneously, rather than memory-based LUT, register array is preferred. In addition, to implement the desired digital FIR filter, fewer cycles are required to update the contents presented in the register based LUT compared to the memory based LUT.

The RPPG partial products are generated based on the MUXes. In this case RPPG outputs are 8-bits each. Each 1-bit output from the RPPG is fed into the Half-Sum Half-Carry adder. This adder will sum the bits from all the RPPGs and 8 outputs will be generated in this example. All the outputs are fed into the Pipeline Shift adder. Using the following equation the final output will be calculated using partial products. Pipeline shift adder will generate output1 to ouput7; among these any one output may be the final output. Because sometimes any output may become zero, then equations get terminated.

$$P_n = \frac{P_{n-1}}{2} + (TF \times 2^{n-1})$$

Where P_n is Partial product
Pn-1 is previous partial product
TF is Twiddle factor

Here n=4, because input bits 8,4,3,2, are in the range of four bit patterns as input.

The number of coefficients and inputs are increased corresponding binary reorientation also increases.

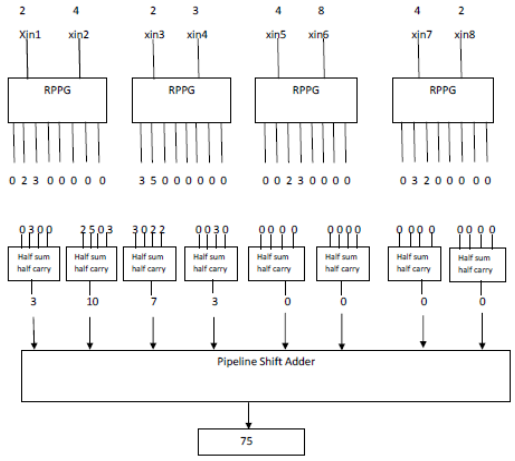

Fig. 2: Proposed block diagram

A. *Numerical Example*

1) Theory:

Input 1----->0 0 0 0 0 0 1 0 - 2
Input 2----- >0 0 0 0 0 1 0 0 - 4
Input 3----- >0 0 0 0 0 0 1 0 - 2
Input 4----- >0 0 0 0 0 0 1 1 - 3
Input 5----- >0 0 0 0 0 1 0 0 - 4
Input 6----- >0 0 0 0 1 0 0 0 - 8
Input 7----- >0 0 0 0 0 1 0 0 - 4
Input 8----- >0 0 0 0 0 0 1 0 - 2

2) DA matrix:

3 2 3 2 3 2 3 2
0 0 0 0 1 0 0 0----- >3
1 0 0 0 1 1 0 1----- >10
0 1 0 1 0 0 1 0----- >7
0 0 1 0 0 0 0 0----- >3
0 0 0 0 0 0 0 0----- >0
0 0 0 0 0 0 0 0----- >0
0 0 0 0 0 0 0 0----- >0
0 0 0 0 0 0 0 0----- >0

3) Multiplication:

2 4 2 3 4 8 4 2
2 3 2 3 2 3 2 3
---
4 12 4 9 8 24 8 6
4 + 12 + 4 + 9 + 8 + 24 + 8 + 6 = 75

- For 8 filter coefficients there are 256 various binary representations. Eight filter coefficients grouped as four, RPPG=0, RPPG=1, RPPG=2 and RPPG=3.
- Now, 2 coefficients will have only 4 binary representations (complexity reduced with RPPGs).
- 0 0
- 0 1
- 1 0
- 1 1
- However 0 0 combinations hold zero value. (Technically only three combinations only).
- Each combination holds corresponding value according to DA-Algorithm and stored in memory.
- RPPG selects any one value for selected binary combination (Each RPPG having one MUX which selects).
- RPPG=0, RPPG=1, RPPG=2 and RPPG=3 have four values get added by Parallel Pipeline adder.

IV. RESULTS

A. *ASIC*

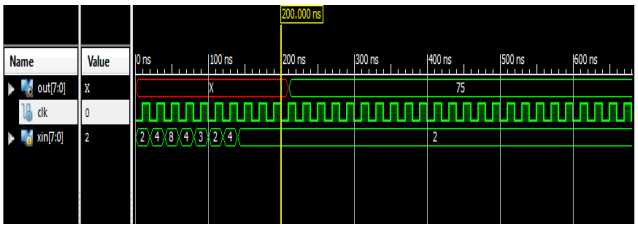

Fig. 3: Simulation of proposed DA – based Reconfigurable FIR digital filter for ASIC implementation.

B. *RPPG Result*

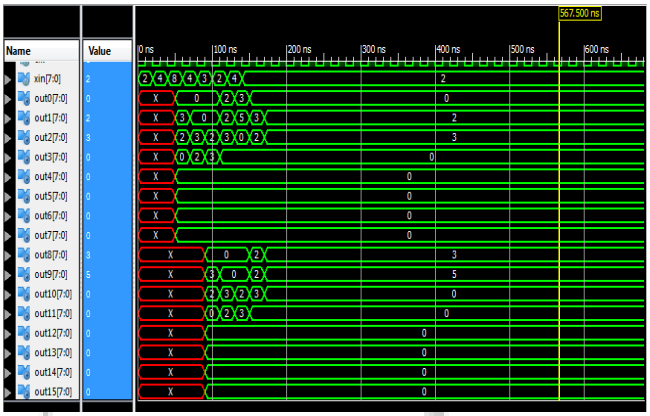

Fig. 4: RPPG is the partial product generator of the coefficients of each 2 bits.

C. *PPA*

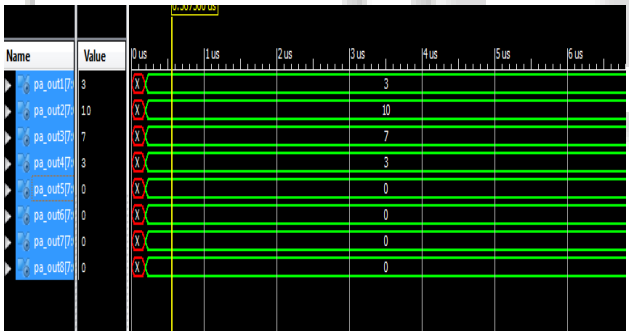

Fig. 5: Parallel pipe line adder

Results comparison table for DA-based systolic structure, DA-based structure using Carry Save Adder and DA-based Reconfigurable FIR digital filter is given in the following Table 1.

Design	MSP	MSF	Area	ADP	EPS
DA-based systolic structures	1.79	558	71195	127439	60.76
DA-based structure using carry save adder (CSA)	5.20	192	18257	94936	32.38
DA-based Reconfigurable FIR Digital filter using Half-Sum-Half-Carry (HSHC)	1.58	632	25163	39757	13.03

Table: 1 Performance Comparison of ASIC Synthesis Results for L = W = 8, M =2, and N =16

Units: MSP (ns), MSF (MHz), Area (sq.um), ADP (sq.umxns), and EPS (mWxns)

V. SUMMARY AND CONCLUSION

This paper presents the design and implementation of DA-based reconfigurable FIR digital filter design. Using this DA-based reconfigurable implementation of FIR digital filter, we can achieve high-throughput and for the given Serial input, Parallel output can be achieved. It is shown that by sharing the same registers by the DA units for different bit slices, the cost of hardware could be reduced substantially.

REFERENCES

[1] Y.Wang, J. Ostermann, and Y. Zhang, Video Processing and Communications, 1st ed. Englewood Cliffs, NJ: Prentice-Hall, 2002.

[2] Y. Chang and C.Wang, "New systolic array implementation of the 2-D Discrete cosine transform and its inverse," IEEE Trans. Circuits Syst.Video Technol., vol. 5, no. 2, pp. 150–157, Apr. 1995.

[3] C. T. Lin, Y. C. Yu, and L. D. Van, "Cost-effective triple-mode reconfigurable Pipeline FFT/IFFT/2-D Vedic Multiplier processor," IEEE Trans. Very Large Scale Integr. Syst., vol. 16, no. 8, pp. 1058–1071, Aug. 2008.

[4] S. Uramoto, Y. Inoue, A. Takabatake, J. Takeda, Y. Yamashita, H. Yerane, and M. Yoshimoto, "A 100-MHz 2-D discrete cosine transform Core processor," IEEE J. Solid-State Circuits, vol. 27, no. 4, pp. 492–499, Apr. 1s992.

[5] M. M. Peiro, E. I. Boemo, and L. Wanhammar, "Design of high-speed multiplier less filters using a nonrecursive signed common sub expression algorithm," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., vol. 49, no. 3, pp. 196–203, Mar. 2002.

[6] C.-H. Chang, J. Chen, and A. P. Vinod, "Information theoretic approach to complexity reduction of FIR filter design," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 55, no. 8, pp. 2310–2321, Sep. 2008.

[7] F. Xu, C. H. Chang, and C. C. Jong, "Contention resolution—A new approach to versatile sub expressions sharing in multiple constant multiplications," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 55, no. 2, pp. 559–571, Mar. 2008.

[8] F. Xu, C. H. Chang, and C. C. Jong, "Contention resolution algorithms for common subexpression elimination in digital filter design," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 52, no. 10, pp. 695–700, Oct. 2005.

[9] I.-C. Park and H.-J. Kang, "Digital filter synthesis based on an algorithmto generate all minimal signed digit representations," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 21, no. 12, pp. 1525–1529, Dec. 2002.