Area Efficient and Low Power Reconfiurable Fir Filter

A. UMASANKAR
Research scholar St.peter's university, ECE, Chennai- 600054, INDIA

N.VASUDEVAN
Dean (Engineering and Technology) , SRM University, Chennai - 600026,INDIA

N.Kirubanandasarathy
Assistant Professor, Syed Ammal Engineering College, Ramanathapuram – 623502, INDIA

Summary
PSM architecture synthesizes multiplier blocks with low hardware requirement suitable for implementation as part of full parallel finite impulse response (FIR) filters is presented in this paper. FIR digital filters are widely used in DSP by the virtue of its stability, linear phase, fewer finite precision error and efficient implementation. In this paper, new reconfigurable architectures of low complexity FIR filters are proposed, namely programmable shifts method. The Methods can be modified by replacing the adder architecture by using carry save adder instead of normal adder architecture. The proposed architectures offer 12% of area and power reductions and compared to the best existing reconfigurable FIR filter implementations in the literature and the proposed architectures have been implemented and tested on Spartan-3 xc3s200-5pq208 field-programmable gate array (FPGA) and synthesized.

Keywords
Channelizer, FIR filter, high level synthesis, CSA, reconfigurability

I. Introduction

Finite impulse response (FIR) digital filters find extensive applications in mobile communication systems for applications such as channelization, channel equalization, matched filtering, and pulse shaping, due to their absolute stability and linear phase proper-ties. The filters employed in mobile systems must be realized to consume less power and operate at high speed. Recently, with the advent of software defined radio (SDR) technology, finite impulses response (FIR) filter research has been focused on reconfigurable realizations. The fundamental idea of an SDR is to replace most of the analog signal processing in the transceivers with digital signal processing in order to provide the advantage of flexibility through reconfiguration. This will enable different air-interfaces to be implemented on a single generic hardware platform to support multi standard wireless communications [1]. Wideband receivers in SDR must be realized to meet the stringent specifications of low power consumption and high speed. Reconfigurability of the receiver to work with different wireless communication standards is another key requirement in an SDR. The most computationally intensive part of an SDR receiver is the channelizer since it operates at the highest sampling rate [2]. It extracts multiple narrowband channels from a wideband signal using a bank of FIR filters, called channel filters.

II. Minimized adder graph algorithm

The Minimized Adder Graph (MAG) algorithm must be introduced [8]. Minimized Adder Graph (MAG) was defined by Dumpster and Macleod and generates minimum adder graphs consisting of shifts, adds and subtracts for implementing constant integer multiplication of individual values up to12-bits (MAG was further extended to 19-bits by Gustafsonal.). Multiplication by a constant is common when implementing DSP on FPGAs, and efficient implementation has received attention from the research community. For example, Wirthlin and McMurtrey [5] utilize features of recent FPGA architectures to further reduce the area requirement of conventional constant coefficient multipliers. The implementation techniques described also allow for multiplication constants to be changed since it is not the structure of the logic that dictates multiplication values (as is the case with Minimized Adder Graph) but rather look-up table contents. Using the Minimized Adder Graph (MAG) algorithm, Dempster and Macleod demonstrate a 16% average reduction in adder cost over CSD representation and show CSD to reduce average adder cost by 33% over binary representation. From an FPGA perspective, for single coefficients, these adder reductions also translate into hardware savings. In general, for each integer value in range, Minimized Adder Graph (MAG) finds numerous graphs for each value that all implement the required multiplication with the minimum number of adders/ vertices. Dempster and Macleod suggest differentiating between these graphs by calculating the number of single-bit full adders required to implement each graph and selecting the graph requiring the least. This differentiation is motivated by attempting to minimize VLSI area consumed when implementing the multiplication. Minimized Adder Graph (MAG) also generates a table containing the adder cost of each integer value in range

Manuscript received August 5, 2015
Manuscript revised August 20, 2015

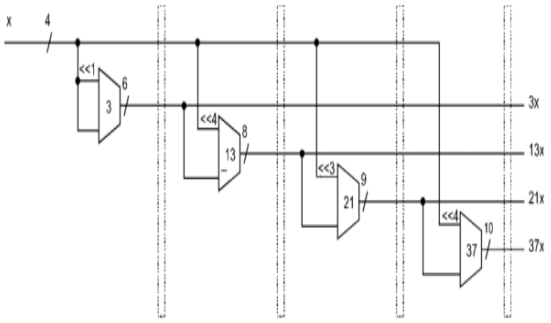

Figure-1: Block Diagram of Minimized Adder Graph

III. Fir Filter With Multiplier Block

Figure-1 shows three full-parallel, fixed-coefficient FIR filter structures that are mathematically identical but differ in architecture. Derived from the standard FIR structure using cut-set retiming, the transposed FIR yields an identical mathematical response but with several advantages for FPGA implementation:

1. No input sample shift registers are required since each sample is fed to each tap simultaneously
2. The pipelined addition chain maps efficiently
3. Filter latency is reduced
4. Identical tap coefficient magnitudes can share multiplication hardware because taps receive the input sample simultaneously.

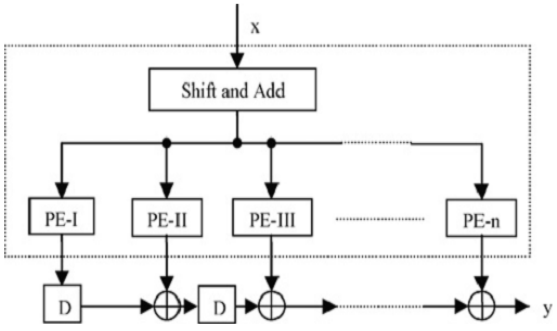

Figure-2: Transposed direct form of an FIR filter

In the transposed direct form, the coefficient multipliers (shown as dotted outline in Figure-1) share the same input and hence commonly known as Multiplier Block (MB). The Multiplier Block (MB) reduces the complexity of the FIR filter implementations, by exploiting the redundancy in MCM. Thus, redundant computations (partial product additions in the multiplier) are eliminated using BCSE. The BCSE method in was formulated as a low complexity solution to realize application specific filters where the coefficients are fixed. In the case of channel filters for Software Defined Radio (SDR) receivers, the coefficients need to be changed as the filter specification changes with the communication standard. Therefore, reconfigurability is a necessary requirement for Software Defined Radio (SDR) channel filters. In the next section, we propose two architectures that incorporate reconfigurability into the BCSE-based low complexity filter architecture. Although we use BCSE to illustrate proposed reconfigurable filter architectures

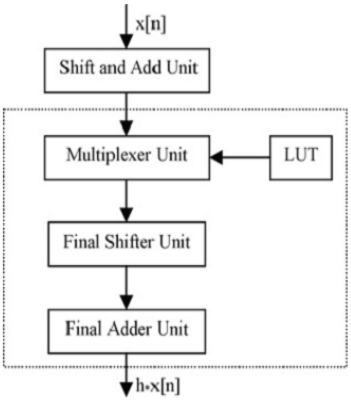

Figure-3: Architecture of the proposed method

IV. Proposed Filter Architectures

In this section, the architecture of the proposed FIR filter is presented. Our architecture is based on the transposed direct form FIR filter structure as shown in Figure-2. The dotted portion in Figure-1 represents the MB. In Figure-3, PE- i represents the processing element corresponding to i Th coefficient. PE performs the coefficient multiplication operation with the help of a shift and add unit which will be explained in the latter part of this section. The architecture of PE is different for proposed MSG and PSM. In the MSG, the filter coefficients are partitioned into fixed groups and hence the PE architecture involves constant shifters. But in the PSM, the PE consists of Programmable Shifters (PS).The functions of different blocks of the PE are explained below.

1) Shift and Add Unit:

It is well known that one of the efficient ways to reduce the complexity of multiplication operation is to realize it using shift and add operations. In contrast to conventional shift and add units used in previously proposed reconfigurable filter architectures, we use the BCSs-based shift and add unit in our pro-posed MSG and PSM architectures. The architecture of shift and add unit is shown in Figure-3. The shift and add unit is used to realize all the 3-bit BCSs of the input signal ranging from [0 0 0] to [1 1 1]. In Figure-3, “x >>k” represents the

input x shifted right by k units. All the3-bit BCSs [0 1 1], [1 0 1], [1 1 0], and [1 1 1] of a 3-bit number are generated using only three adders, whereas a conventional shift and add unit would require five adders. Since the shifts to obtain the BCSs are known beforehand, PS is not required. All these eight BCSs (including [000]) are then fed to the multiplexer unit. In both the architectures (MSG and PSM) proposed in this paper, we use the same shift and add unit. Thus, the use of 3-bit BCSs reduces the number of adders needed to implement the shift and add unit compared to conventional shift and add units.

2) Multiplexer Unit:

The multiplexer units are used to select the appropriate output from the shift and add unit. All the multiplexers will share the outputs of the shift and add unit. The inputs to the multiplexers are the 8/4 inputs from the shift and add unit and hence 8:1/4:1 multiplexer units are employed in the architecture. The select signals of the multiplexers are the filter coefficients which are previously stored in a look up table (LUT). The MSG and PSM architectures basically differ in the way filter coefficients are stored in the LUT. In the MSG, the coefficients are directly stored in LUTs without any modification whereas in PSM, the coefficients are stored in a coded format. The number of multiplexers will also be different for PSM and MSG. In MSG, the number of multiplexers will be dependent on the number of groups after the partitioning of the filter coefficient into fixed groups. The number of multiplexers in the PSM is dependent on the number of non-zero operands in the coefficient for the worst case after the application of BCSE algorithm.

3) Final Shifter Unit:

The final shifter unit will perform the shifting operation after all the intermediate additions (i.e., intra-coefficient additions) are done. This can be illustrated using the output expression shift and add unit is shown in Figure-3. The shift and add unit is used to realize all the 3-bit BCSs of the input signal ranging from [0 0 0] to [1 1 1]. In Figure-3, "x >>k" represents the input x shifted right by k units. All the3-bit BCSs [0 1 1], [1 0 1], [1 1 0], and [1 1 1] of a 3-bitnumber are generated using only three adders, whereas a conventional shift and add unit would require five adders. Since the shifts to obtain the BCSs are known beforehand, PS is not required. All these eight BCSs (including [000]) are then fed to the multiplexer unit. In both the architectures (MSG and PSM) proposed in this paper, we use the same shift and add unit. Thus, the use of 3-bit BCSs reduces the number of adders needed to implement the shift and add unit compared to conventional shift and add units.

4) Final Adder Unit:

This unit will compute the sum of all the intermediate additions $2^{-4}(x + 2^{-2}x)$ and $2^{-15}(x + 2^{-1}x)$ as in [2]. As the filter specifications of different communication standards are different, the coefficients change with the standards. In conventional reconfigurable filters, the new coefficient set corresponding to the filter specification of the new communication standard is loaded in the LUT. Subsequently, the shift and add unit performs a bitwise addition after appropriate shifts. On the contrary, the proposed PSM architectures perform a binary common sub expression (BCS)-wise addition (instead bitwise addition). Thus, the same hardware architecture can be used for different filter specifications to achieve the necessary reconfigurability. Moreover, the proposed BCS-based shift and add unit reduces addition operations and hence offers hardware complexity reduction. Architecture can be used for different filter specifications to achieve the necessary reconfigurability. Moreover, the proposed BCS-based shift and add unit reduces addition operations and hence offers hardware complexity reduction. In the next section, the MSG is explained in a detailed manner.

V. Architecture of PSM

The PSM architecture presented in this section incorporates reconfigurability into BCSE. The PSM has a pre-analysis part in which the filter coefficients are analyzed using the BCSE algorithm in [6]. Thus, the redundant computations (additions) are eliminated using the BCSs and the resulting coefficients in a coded format are stored in the LUT. The coding format is explained in the latter part of this section. The shift and add unit is identical for both PSM and MSG. The number of multiplexer units required can be obtained from the filter coefficients after the application of BCSE [6]. The number of multiplexers is selected after considering the number of non-zero operands (BCSs and unpaired bits) in each of the coefficients after the application of the BCSE algorithm. The number of multiplexers will be corresponding to the number of non-zero operands for the worst-case coefficient (worst-case coefficient being defined as coefficient that has the maximum number of non-zero operands).

The architecture of PE for PSM [6] is shown in Figure-5; the coefficient word length is fixed as 16 bits. We have done the statistical analysis for various filters with coefficient precision of 16 bits and different filter lengths (20, 50, 80, 120, 200,400, and 800 taps) and it was found that the maximum number of non-zero operands is 5 for any coefficient. The analysis was done for filters with different pass band (ωp) and stop band (ωs) frequency specifications given by

$\omega p = 0.1\pi, \omega s = 0.12\pi$
$\omega p = 0.15\pi, \omega s = 0.25\pi$
$\omega p = 0.2\pi, \omega s = 0.22\pi$
$\omega p = 0.2\pi, \omega s = 0.3\pi$

Based on our statistical analysis, we have fixed the number of multiplexers as 5 (same as the number of non-zero operands). The LUT consists of two rows of 18 bits for each coefficient of the form SDDDDXXDDDDXXMMMMLandDDDDXXDDDDX XDDDDXX, where "S" represents the sign bit, "DDDD" represents the shift values from 20 to 2−15and "XX" represents the input "x" or the BCSs obtained from the shift and add unit. In the coded format, XX ="01" represents "x" "10" represents x +2−1x, "11" represents x +2−2x, and "00" represents x +2−1 x +2−2x, respectively. Thus, the two rows can store up to five operands which is the worst case number of operands for a 16-bit coefficient. In most of the practical coefficients, the number of operands is less than the worst case number of operands, In that case "MMMML" can be used to avoid unnecessary additions. The values "MMMM" will be given as select signal to the Mux6and "L" to Mux8. "MMMML" indicates the presence of five operands.

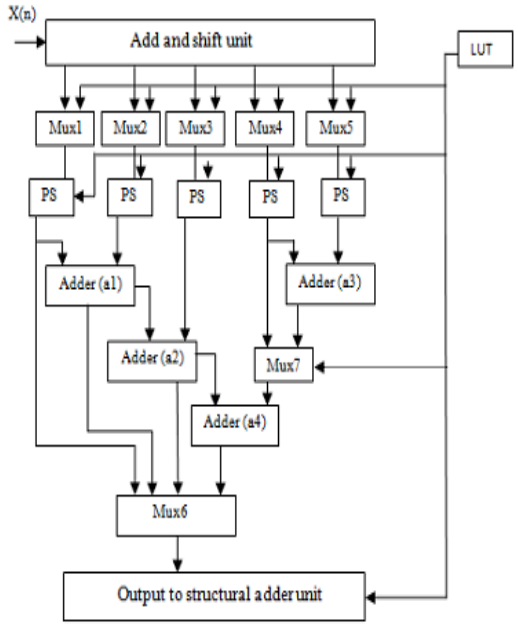

Figure-4: Architecture of PE for PSM

VI. Carry save adder

Basically, carry save adder is used to compute sum of three or more n-bit binary numbers. Carry save adder is same as a full adder. But as shown in figure-4[7], here we are computing sum of two 16-bit binary numbers, so we take 16 half adders at first stage instead of using 16 full adders. Therefore, carry save unit consists of 16 half adders, each of which computes single sum and carry bit based only on the corresponding bits of the two input numbers.

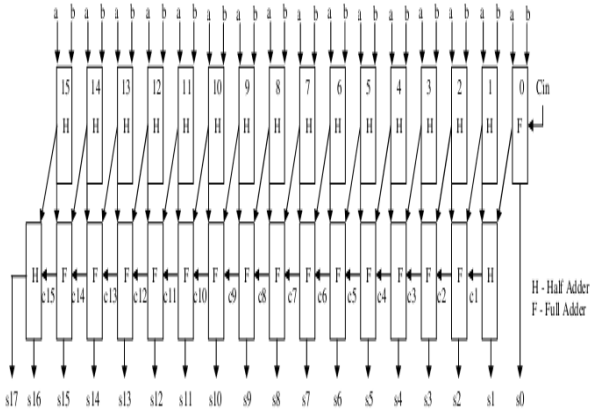

Figure-5: Block Diagram of 16-bit Carry Save Adder

VII. Results & Simulation

The results presented establish a clear area advantage of PSM over prior algorithms For Typical filter parameters with comparable maximum clock rates. In addition, the Industrial Relevance of the transposed FIR with multiplier block architecture and the RSG algorithm has been established through comparison with filters implemented using the Distributive Arithmetic Technique. In this chapter we presented a programmable multiplier less technique, based on the add and shift method and common sub expression elimination for low area, low Power and high speed implementations of FIR filters. We validated our techniques on Spartan-III devices where we observed significant area and power reductions over traditional Distributed Arithmetic based techniques and multiplier less technique.

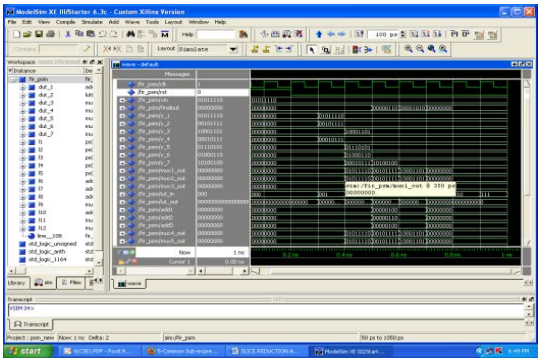

Figure-6: Screen shot showing the simulation of PSM with filter

Table 1.Comparison table of Power and Area of MSG and PSM

Methods	MSG	PSM
Slices	32	20
LUT	41	14
Power	0.389	0.268

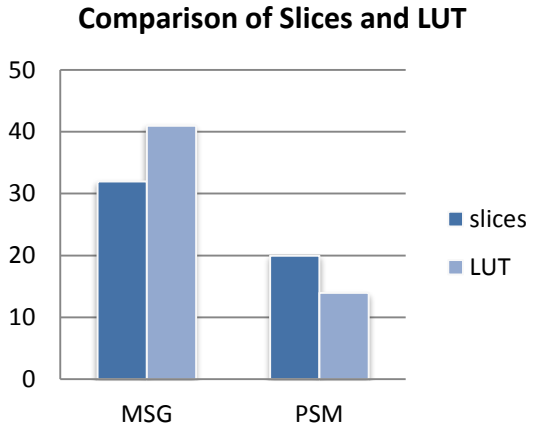

Figure-7: Performance comparison of MSG and PSM

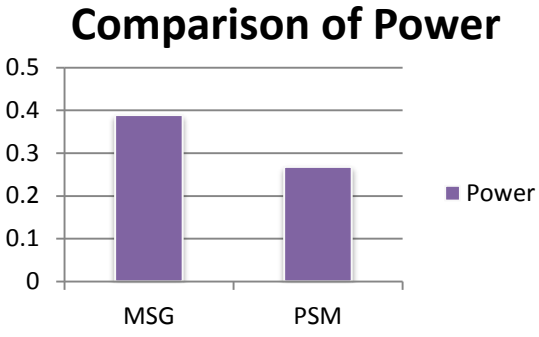

Figure-8: Comparison of Power in MSG and PSM Methods

Conclusion

We have proposed two new approaches namely, MSG and PSM, for implementing reconfigurable higher order filters with low complexity. The MSG architecture results in high speed filters and PSM architecture results in low area and thus low power filter implementations. The PSM also provides the flexibility of changing the filter coefficient word lengths dynamically. We have implemented the architectures on Spartan-III XC3S200-5PQ-208 FPGA and synthesized. The proposed reconfigurable architectures gave low Power and Area.

References

[1] T. Hentschel and G. Fettweis, “Software radio receivers,” in CDMA Techniques for Third Generation Mobile Systems, Dordrecht, The Netherlands: Kluwer Academic, 1999, pp. 257–283

[2] J. Mitola, “Object-oriented approaches to wireless systems engineering," in Software Radio Architecture. New York: Wiley, 2000.

[3] Muhammad, Khurram and Roy, Kaushik, "A Computational Redundancy Reduction Approach for High performance and Low Power DSP Algorithm Implementation" (1999). ECE Technical Reports Paper 36

[4] B. Parhami, “Implementation details,” in Computer Arithmetic. New York: Oxford Press, 2000, p. 131.

[5] X.-J. Zhang, K.-W. Ng, and W. Luk, “A combined approach to high level synthesis for dynamically reconfigurable systems,” in Proc. 10th Int. Workshop Field Programmable Logic Applicat., 2000, pp. 361–370

[6] R. Mahesh, Member, IEEE, and A. P. Vinod, Senior Member, IEEE “New Reconfigurable Architectures for Implementing FIR Filters with Low Complexity” IEEE transactions on computer-aided design of integrated circuits and systems, vol. 29, no. 2, February 2010.

[7] B.Ramkumar, Harish M Kittur, P.Mahesh Kannan” ASIC Implementation of Modified Faster Carry Save Adder” European Journal of Scientific Research ISSN 1450-216X Vol.42 No.1 (2010), pp.53-58

[8] K.N. Macpherson and R.W. Stewart” Area efficient FIR filters for high speed FPGA Implementation” RAPID PROTOTYPING, Dec 2006.

[9] M. M. Peiro, E. I. Boemo, and L. Wanhammar, “Design of high-speed multiplier less filters using a non recursive signed common sub expression algorithm,” IEEE Trans. Circuits Syst. II , vol. 49, no. 3, pp. 196203,Mar. 2002.

[10] R. Mahesh and A. P. Vinod, “A new common sub expression elimination algorithm for realizing low complexity higher order digital filters,” IEEE Trans. Computer Aided Design Integrated Circuits Syst. , vol. 27, no. 2, pp.217–219, Feb. 2008.