Advances in Computer Science and Information Technology (ACSIT)
p-ISSN: 2393-9907; e-ISSN: 2393-9915; Volume 2, Issue 14; October-December, 2015 pp. 59-62

http://www.krishisanskriti.org/Publication.html

A New Modified Reducible Complexity Hardware Efficient FIR Filter by Using Proposed Shift and Adder Architectures

P. Hemalatha[1] and Sateesh Reddy[2]

[1,2]UCET, Guntur
E-Mail: [2]hemasashidhar@gmail.com

Abstract—*Finite-impulse response (FIR) Filter is widely used in wireless sensor networks as a signal pre-processing step. Because sensor nodes require a long working periods and ultra-low cost, traditional FIR structures are inapplicable as multipliers occupy too much die size for such node's chips. This paper proposes novel FIR filter structures used in the design of application specific integrated circuits (ASICs) for sensor nodes, and to add with again my project includes reconfigurable FIR filters using CSM and PSM which can reduce the hardware cost to a minimum by giving reconfig. ability and reduce the complexity. The experiments show that the proposed FIR structure can lead to significant hardware savings from the traditional FIR filter. This architecture as the capacity to operate over varies word length coefficient of the filter. also, it easy to prove that this will offer good area, improved speed & power reduction than the existing one.*

Keywords*: DSP, ASICs, FIR, VLSI, CSM, PSM*

1. INTRODUCTION

FIR digital filters are one of the most widely used fundamental devices performed in digital signal processing systems, ranging from video and image processing to wireless communication. Many efforts have been directed to wireless communication. In, an approach to increase the throughput of FIR filters with reduced complexity hardware is presented. This approach starts with the short convolution algorithms, which are transposed to obtain computationally efficient parallel filter structures. In poly phase decomposition and fast FIR algorithms (FFA) are used to implement parallel FIR filter. The FFAs are iterated to get fast parallel FIR algorithms for larger block sizes. In parallel FIR sub-filters are implemented by a set of fast block filtering algorithms which are based on fast short length liner convolution algorithms. However, the number of additions will increase along with the convolution length, which will lead to complex pre-addition and post-addition matrices that are not practical for hardware implementation for wireless sensors. In,Cook-Toom algorithm and Winograd algorithm are put forward for fast convolution of any filter length, whereas their pre-addition and post-addition matrices may contain elements that are not in the set {-1,0,1},which makes it not suitable for hardware implementation either. In, a new hardware efficient fast parallel FIR filter structure is obtained by a linear convolution structure based on iterated short convolution algorithm, which can save a large amount of hardware cost. In the approach based on poly phase decomposition, and additional delay elements are integrated into the post-addition matrix, when the block sizes become large, these designs are irregular and require large amount of delay elements. In, the fast liner convolution is used to develop the small-sized filtering structures and larger block-sized filtering structures are constructed through iterations of the small-sized filtering structures. In [10], a new parallel FIR filter structure is presented, which exploits the inherent nature of symmetric coefficients and reduces half multipliers in sub-filter section at the expense of additional adders in preprocessing and post processing blocks. However, only under the condition that the number of taps is a multiple of 2 or 3 can the filter structure be beneficial. In [11], a computation reduction technique called computation sharing differential coefficient (CSDC) method is presented, which can be used to obtain low-complexity multiplier less implementation of FIR filters

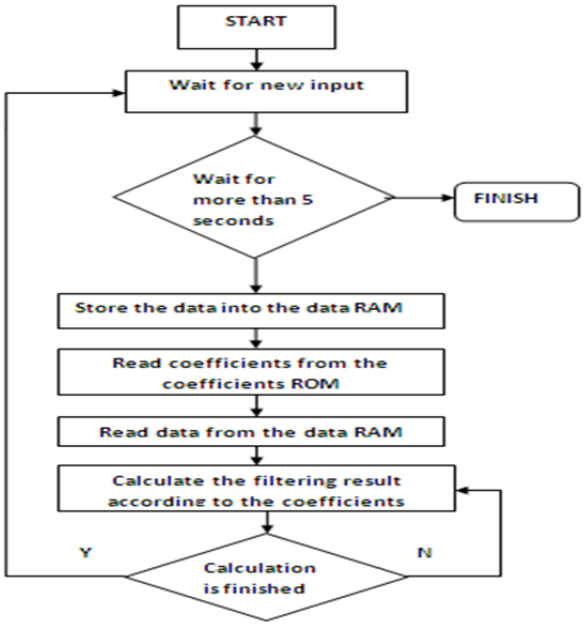
2. FIR ALGORITHM

An n-tap FIR filter can be expressed in the general form as (1),

$$Y(n) = \sum_{i=0}^{N-1} \mathrm{h(i)x(n-i)}, \mathrm{n} = 0,1,2,\ldots\ldots\infty \rightarrow 1$$

where x(n) is an infinite-length input sequence and h(i) are the coefficients of the length-N FIR filter. N multipliers and N-1 adders are needed to implement an N-tap FIR filter. And according to, the traditional L-parallel FIR filter can be derived using poly phase decomposition, and we can get as follows.

$$\sum_{p=0}^{L-1} Yp(z^L)\mathrm{z}^{-\mathrm{p}} = \sum_{p=0}^{L-1} Yp(z^L)\mathrm{z}^{-\mathrm{p}} = \sum_{q=0}^{L-1} Hr(z^L)\mathrm{z}^{-\mathrm{r}}$$



$$Xp = \sum_{k=0}^{\infty} z^{-k}\mathrm{x}(\mathrm{Lk}+\mathrm{q})$$

$$Yp = \sum_{k=0}^{\infty} z^{-k}\mathrm{x}(\mathrm{Lk}+\mathrm{p})$$

$$Hr = \sum_{i=0}^{N/L-1} z^{-k}\mathrm{x}(\mathrm{Lk}+\mathrm{r}), \mathrm{p}, \mathrm{q}, \mathrm{r} = 0,1,2,\ldots\ldots \mathrm{L}-1$$

We can fig. out that the traditional L-parallel FIR filter will need L2 FIR sub-filters of length N/L, which means LN multipliers and L (N-1) adders are required for hardware implementation. Notice that our goal is to reduce the hardware cost and energy consumption to a minimum, whereas an L-parallel FIR filter will increase the hardware cost, which makes it an improper option of implementing FIR filters in wireless sensor network. Taking no account of processing speed, a simple way to implement FIR filters requires N multipliers and (N-1) adders. Processing speed does not play a pivotal role in wireless sensor networks. What we care about most is how cheap the node is and how long the node can work, not how fast we can get the results. As we all know, the phenomena of optimum processing speed and optimum hardware cost can't appear simultaneously. The reduction in hardware cost will definitely affect the processing speed of FIR filters. Obviously, it's a multi-objective optimization problem. But in this specific situation—wireless sensor network, it's reasonable to sacrifice a certain degree of processing speed for achieving the optimum hardware cost. Without doubt, the processing speed is tolerated and we can ensure that the system will not fail because of cannot get the filter results in time. Therefore, we propose a novel FIR filter structure which can be used in the design of application specific integrated circuits (ASICs) for sensor nodes for the reason that the filter structure can reduce the hardware cost to a minimum.

3. PROPOSED ALGORITHM

To reduce the hardware cost to a minimum, the main idea behind the proposed FIR algorithm is actually pretty intuitive to exchange multipliers and adders with shifters as shifters weigh less than multipliers and adders in terms of silicon area. All we need are an adder and a shifter to implement a FIR filter. Therefore, for an N-tap FIR filter the total amount of saved multipliers would be N, that is we do not need multipliers at all, the total amount of saved adders would be N-2, at the expense of one additional shifter, which occupies much less hardware resource and cut off die size than multipliers and adders.

The proposed FIR algorithm will be presented in the flowing. First, encode all the coefficients of the length-N FIR filter in a binary representation with fixed point format. To make it easier to understand, an example is given in the following

Assuming that the coefficients of an 8-tap FIR filter are 0.3759, 0.3086, -0.1255, 0.06439, 0.2187, 0.0168, -0.00415 and 0.01758. And then their binary representations can be expressed as

h(0) = 0.3759 = 2-1+(-2-3)+2-10

h(1) = 0.3086 = 2-2+2-4+(-2-8)

h(2) = -0.1255 = -2-3+(-2-11)

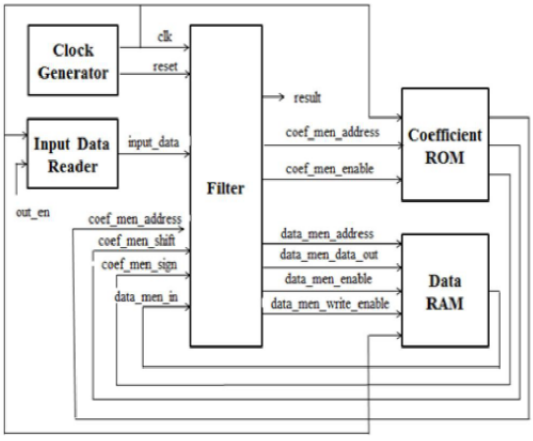
h(3) = 0.6439 * 10-1= 2-4+2-9

h(4) = 0.2187 = 2-2 +(-2-5)

h(5) = 0.0168 = 2-6+2-10+2-12

h(6) = -0.415 * 10-2 = -2-8+(-2-12)

h(7) = 0.1758 * 10-1 = 2-6+2-9



According to (1), an 8-tap FIR filter's expansion can be expressed as

$$Y(n) = h(0) * x(n) + h(1) * x(n-1) + h(2) * x(n-2) + h(3) * x(n-3) + h(4) * x(n-4) + h(5) * x(n-5) + h(6) * x(n-6) + h(7) * x(n-7)$$

The coefficients mentioned before are substituted into (3). And we get can get (4) as shown in the following.

$$y(n) = (2^{-1} - 2^{-3} + 2^{-10}) * \mathrm{x(n)} + (2^{-2} - 2^{-4} + 2^{-8}) * x(n-1) + (-2^{-3} - 2^{-11}) * x(n-2) + (2^{-4} + 2^{-9}) * x(n-3) + (2^{-2} + 2^{-5}) * x(n-4) + (2^{-6} - 2^{-10} + 2^{-12}) * x(n-5) + (2^{-8} + 2^{-12}) * x(n-6) + (2^{-6} + 2^{-9}) * x(n-7)$$

Obviously, all the inputs should be multiplied by a sequence of numbers that are specific powers of 2, what's more, some inputs should be multiplied by the same number, for instance, both x(n) and x(n-2) have to multiply by -2-3, so we can get all the inputs that have to multiply by the same number together. In this way, we can get (5), which is shown as follows.

$$y(n) = 2^{-1} * x(n) + 2^{-2} * \big(x(n-1) + x(n-4)\big) + x^{-3} \big(-x(n) - x(n-2)\big) + 2^{-4} * \big(x(n-1) + x(n-3)\big) - 2^{-5} * x(n-4) + 2^{-6} * \big(x(n-5) + x(n-7)\big) - 2^{-8} * x(n-1) + 2^{-9} * \big(x(n-3) + x(n-7)\big) + 2^{-10} * \big(x(n) + x(n-5)\big) - 2^{-11} * x(n-2) + 2^{-12} * (x(n-5) - x(n-6))$$

$$y(n) = 2^{-1} * (\ldots.2(2 * U1 + U2) + U3) + \cdots . + U12$$

where U1 = x(n-5)-x (n-6); U2 = -x (n-2);

U3 = x (n) +x (n-5); U4 = x (n-3) +x (n-7);

U5 = -x (n-1); U6 = 0;

U7 = x (n-5) +x (n-7); U8 = -x (n-4);

U9 = x (n-1) +x (n-3); U10 = - x (n)+(-x (n-2))

U11 = x (n-1) +x (n-4); U12 = x (n);

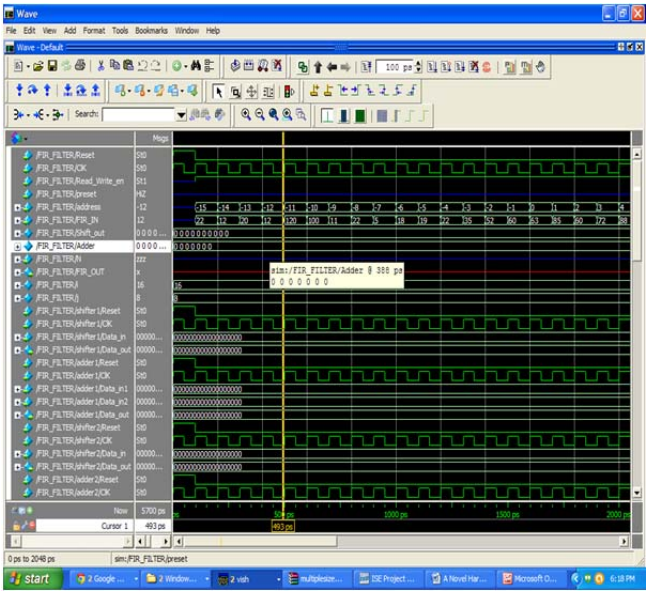
Then, we calculate U1 by adding x(n-5) to -x(n-6), and then multiply U1 by 2-1,which means shift to the right by one bit, that is easy for hardware implementation. And then add the results to U2, shift to the right again, add to U3 which is calculated by adding x(n) to x(n-5), and so on, when we complete all the operations , we can get y(n), the result of FIR filter.

In this way, we can get the filter results by addition and shifting, rather than multiplication. Therefore, there is no need to use multipliers to implement FIR filters at all. All we need are an adder and a shifter.

The hardware cost can be reduced to a minimum in this way. For a conclusion, firstly, present all the coefficients in an N-tap FIR filter's expansion in a binary representation , then gather all the inputs that are supposed to multiply by the same number together, and can be get, according to which we can calculate the filter results by addition and shifting. In this way, all we need are an adder and a shifter in the proposed FIR structures.

Substituting multipliers and adder with shifter has advantages of silicon area and power consumption. In addition, the increased shifter stay fixed when the length of the FIR filters increase. This is the most hardware efficient way to implement FIR filters, which makes it very suitable to be used in sensor node ASICs.

4. PROJECT FLOW

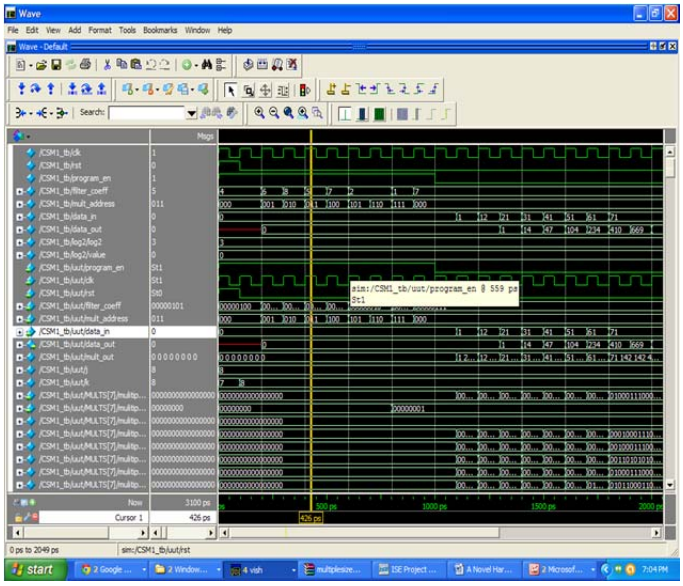


Simulation Result for proposed Fir Filter

A project is a album mechanism for an HDL design underneath pattern or test. Even though you don't have to use projects in ModelSim, they may ease dealings with the tool and are handy for organizing files and specifying simulation settings

Device Utilization Summary				[-]
Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	237	17,344	1%	
Number of 4 input LUTs	1,069	17,344	6%	
Number of occupied Slices	627	8,672	7%	

Number of Slices containing only related logic	627	627	100%	
Number of Slices containing unrelated logic	0	627	0%	
Total Number of 4 input LUTs	1,099	17,344	6%	
Number used as logic	1,063			
Number used as a route-thru	30			
Number used as Shift registers	6			
Number of bonded IOBs	41	250	16%	
Number of BUFGMUXs	1	24	4%	
Average Fanout of Non-Clock Nets	3.74			



SIMULATION RESUKTS FOR CSM

5. FUTURE SCOPE

In this way, we can get the filter results by addition and shifting, rather than multiplication. Therefore, there is no need to use multipliers to implement FIR filters at all. All we need are an adder and a shifter.

The experiment results have shown that the proposed FIR algorithm is very efficient in reducing hardware cost and energy consumption, especially when the length of the fir filter is large. the proposed fir structures can do a better job than the traditional structure in reducing hardware cost. Therefore, the proposed FIR structure is a better choice for sensor node ASICS.

REFERENCE

[1] "Hardware Description Language."Wikepedia:The Free Encyclopedia.18May 2004.http://en.wikipedia.org/wiki/Hardware_description_language.

[2] Xilinx Inc., "Virtex-II Pro□ □Platform FPGAs: Functional Description," DS083- 2 (v3.0), December 10, 2003.

[3] Project Veripage, retrieved from: http://www.angelfire.com/ca/verilog/history.html.

[4] SudhakarYalamanchili, Introductory VHDL,From Simulation to Synthesis, Prentice Hall, 2001.

[5] Y. Wang and K. Roy, "CSDC: A new complexity reduction technique for multiplierless implementation of digital FIR filters," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 52, no. 9, pp. 1845-1853, Sep. 2005.

[6] H. Lee, J. Chung, and G. Sobelman, "FPGA-based digit-serial CSD FIR filter for image signal format conversion," in Proc. Int. Conf. Signal Processing Application Technology (ICSPAT'98), Toronto, ON. Canada, Sept. 1998.

[7] . M. Martinez-Peiro, E. Boemo, and L. Wanhammer, "Design of highspeed multiplierless filters using a nonrecursive signed common subexpression algorithm," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., vol. 49, no. 3, pp. 196-203, Mar. 2002.