International Journal of Electronics Communication and Computer Engineering
Volume 4, Issue 2, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

Design a Novel Based Fir Filter Architecture for High Speed

Mis. M. Anuvidhya
Department of ECE M.E Applied Electronics,
Magna College of engineering, Chennai.
Email: anuvidhya04@gmail.com

Mr. A. M. S. Alaksandar Jesus Gene M.E.,
Department of ECE
Magna College of engineering, Chennai.
Email:alaksandarjesus@yahoo.com

***Abstract* – Finite Impulse Response (FIR) filter which deals with the new algorithm of Reconfigurability and low complexity. The algorithm that synthesizes multiplier block with low hardware requirement suitable for implementation as part of full-parallel FIR filter. Minimizing multiplier block logic depth and pipeline registers is shown to have the greatest influence in reducing FPGA area cost. In addition to providing lower area solutions than existing algorithms, comparisons with equivalent filters generated using the distributed arithmetic technique demonstrate further area advantages of the new algorithm. In the proposed architectures pipeline is introduced so that speed is increased by decreasing the delay. This technique used in the processor in which the data processing part is divided into many stages separated by register. Thus every storage can process one instruction in a clock cycle and the instruction move forward in the pipeline. Thus the processor can execute many instructions in much lesser time.**

***Keywords* – Reconfigurability, Pipe Line, High Level Synthesis, Common Subexpression Elimination, FIR Filter, Channelizer, Constant Shift Method.**

I. INTRODUCTION

1.1 FIR Filter

FIR filters are one of two primary types of digital filters used in Digital Signal Processing (DSP) applications, the other type being IIR. "FIR" means "Finite Impulse Response". If you put in an impulse, that is, a single "1" sample followed by many "0" samples, zeroes will come out after the "1" sample has made its way through the delay line of the filter. The input–output relationship of the linear time invariant (LTI) FIR filter can be described as

$$y(n) = \sum_{k=0}^{M-1} C_k , x(n-k)$$

Where M represents the length of the FIR filter, the clock are the filter coefficients and x(n-k) denotes the data sample at time constant (n-k).

In the common case, the impulse response is finite because there is no feedback in the FIR. A lack of feedback guarantees that the impulse response will be finite. Therefore, the term "finite impulse response" is nearly synonymous with "no feedback". However, if feedback is employed yet the impulse response is finite, the filter still is a FIR. An example is the moving average filter, in which the Nth prior sample is subtracted (fed back) each time a new sample comes in. This filter has a finite impulse response even though it uses feedback: after N samples of an impulse, the output will always be zero.

II. REVIEW OF EXISTING SYSTEM

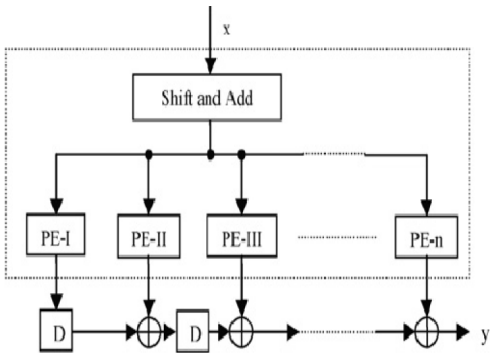

Fig.2.1. Transposed direct form of fir filter

In the Fig 2.1 Architecture is based on the transposed direct form FIR filter structure The dotted portion represents the MB, PE-*i* represents the processing element corresponding to the *i*th coefficient 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 the PE architecture involves constant shifters.

Here the PE consists of programmable shifters (PS). The FIR filter architecture can be realized in a serial way in which the same PE is used for generation of all partial products by convolving the coefficients with the input signal ($h * x[n]$) or in a parallel way, where parallel PE architectures are employed. The first option is used when power consumption and area are of prime concern.

2.1 The functional block of the PE

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 proposed CSM and PSM architectures. The shift and add unit bit BCSs of the input signal ranging from [0 0 0] to [1 1 1]. In Fig. 3, "$x>>k$" represents the input x shifted right by k units. All the 3-bit BCSs [0 1 1], [1 0 1], [1 1 0], and [1 1 1] of a 3-bit 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

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

International Journal of Electronics Communication and Computer Engineering
Volume 4, Issue 2, ISSN (Online): 2249–071X, ISSN (Print): 2278–4209

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 CSM and PSM architectures basically differ in the way filter coefficients are stored in the LUT. In the CSM, 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 CSM. In CSM, 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

Final Shifter Unit*:* The final shifter unit will perform the shifting operation after all the intermediate additions (i.e., intra-coefficient additions) are done. The multiplexer unit, final shifter unit will perform the shift operations In the CSM, the final shifts are constants and hence no PS are required.

Final Adder Unit: This unit will compute the sum of all the intermediate additions $2-4(x + 2-2x)$ and $2-15(x + 2-1x)$. 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 final shifter unit will perform the shift operations 2−1, 2−3, and 2−6. Since these shifts are always constant irrespective of the coefficients, programmable shifters are not required and these shifts can be hardwired. The final adder unit will compute the sum of all the intermediate sums to obtain $h * x[n]$. The coefficient word length is considered as 16 bits. The filter coefficients are stored in the LUT in sign-magnitude form with the MSB reserved for the sign bit. The first bit after the sign bit is used to represent the integer part of the coefficient and the remaining 16 bits are used to represent the fractional part of the coefficient. Thus, each 16-bit coefficient is stored as an 18-bit value in LUT.

Each row in LUT corresponds to one coefficient. Note that only half the number of coefficients needs to be stored as FIR filter coefficients are symmetric. The coefficient values corresponding to 20 to 2−14 are partitioned into groups of three bits and are used as select signals to multiplexers Mux1 to Mux5. i.e., the set (20, 2−1, 2−2) forms the select signal to Mux1 and so on. Since there are 3-bits, eight combinations are possible and hence Mux1 to Mux5 are 8:1 multiplexers. The value corresponding to 2−15 forms the select to a 2:1 multiplexer, Mux6. The output from the *i*th multiplexer is denoted as *ri*. Note that even though we are taking coefficient with values up to a precision of 16 bits, the shifting of 2−1 is done.

2.2 Architecture of PSM

In Fig 2.2 PSM architecture incorporates reconfigure-bility into BCSE. The PSM has a pre-analysis part in which the filter coefficients are analyzed using the BCSE algorithm. 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 PSM. The number of multiplexer units required can be obtained from the filter coefficients after the application of BCSE. 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

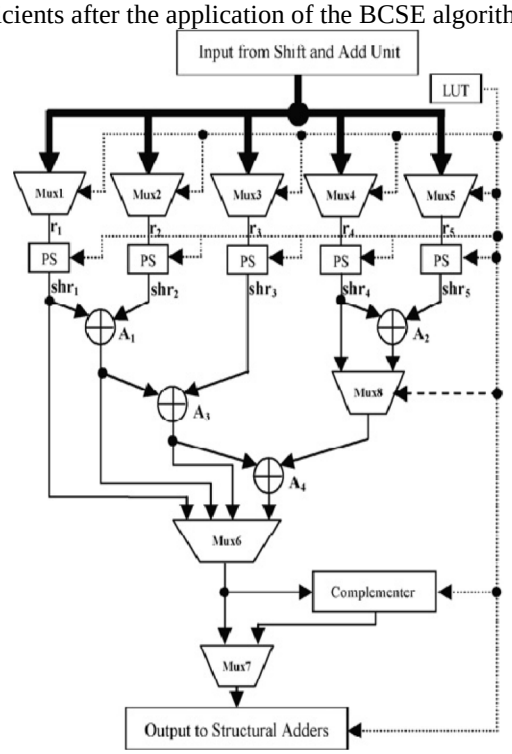

Fig.2.2. Architecture of PE for PSM

III. OVERVIEW OF PROPOSED SYSTEM

Impulse Response (FIR Finite) filters which deals with the new algorithm of Reconfigurability and low complexity. The algorithm that synthesizes multiplier block with low hardware requirement suitable for implementation as part of full-parallel FIR filter. With the PSM block a pipe line is introduced between the series of Mux & PS block. Another pipe line is between the output of Mux 6 & complementer and Mux 7.

As the process take place from the multiplexer, the output is made into the functional block of pipe line .There after the PS block function execute with the corresponding sequence of operation. Though the same process is introduced in the proposed system with the pipe line as additional function towards the block to make the function as faster in the series of block which under goes the steps like Fetch, decode, Store, Execute. The process works in different steps of instruction at the same time, more instructions can be executed in shorter period of time. There by number of instructions can be executed at unit time. Minimizing multiplier block logic depth and pipeline registers is shown to have the greatest influence in reducing FPGA area cost. In addition to providing lower area solutions than existing algorithms, comparisons with

equivalent filters generated using the distributed arithmetic technique demonstrate further area advantages of the new algorithm. In the proposed architectures pipeline is introduced so that speed is increased by decreasing the delay. This technique used in the processor in which the data processing part is divided into many stages separated by register.

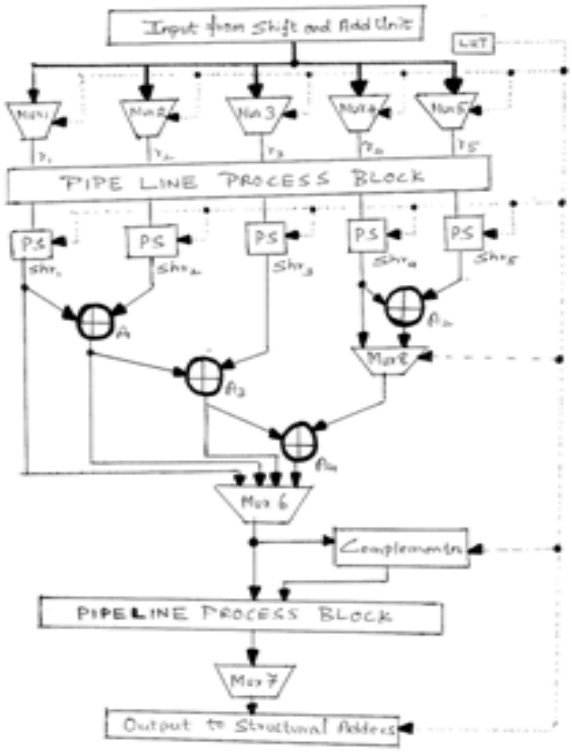

Fig.3.1. Reconfigured fir architecture

Thus every storage can process one instruction in a clock cycle and the instruction move forward in the pipeline. Thus the processor can execute many instructions in much lesser time.

IV. CONCLUSION

In the architectures pipeline is introduced so that speed is increased by decreasing the delay. This technique used in the processor in which the data processing part is divided into many stages separated by register. So every storage can process one instruction in a clock cycle and the instruction move forward in the pipeline. Thus the processor can execute many instructions in much lesser time.

REFERENCES

[1] A. P. Vinod and E. Lai, "Low power and high speed implementation of FIR filters for software defined radio receivers," IEEE Trans. Wireless Commun., vol. 5, no. 7, pp. 1669–1675, Jul. 2006.

[2] C. Y. Yao, H. H. Chen, C. J. Chien, and C. T. Hsu, "A novel common-sub expression-elimination method for synthesizing fixed point FIR filters," IEEE Trans. Circuits Syst. I, vol. 51, no. 11, pp. 2215–2221, Nov. 2004.

[3] I. C. Park and H. J. Kang, "FIR filter synthesis algorithms for minimizing the delay and the number of adders," IEEE Trans. Circuits Syst. II, vol. 48, no. 8, pp. 770–777, Aug 2001

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

[5] 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. 196–203, Mar. 2002.

[6] M. Meribout and M. Motomura, "A combined approach to highlevel synthesis for dynamically reconfigurable systems," IEEE Trans. Comput., vol. 53, no. 12, pp. 1508–1522, Dec. 2004.

[7] P. Tummeltshammer, J. C. Hoe, and M. Puschel, "Multiplexed multiple Constant multiplication," IEEE Trans. Comput.-Aided Design Integr. Circuits, vol. 26, no. 9, pp. 1551–1563, Sep. 2007.

[8] R. Mahesh and A. P. Vinod, "A new common subexpression elimination algorithm for realizing low complexity higher order digital filters," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 27, no. 2, pp. 217–219, Feb. 2008.

[9] R. Pasko, P. Schaumont, V. Derudder, S. Vernalde, and D. Durackova, "A new algorithm for elimination of common sub expressions," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 18, no. 1, pp. 58–68, Jan. 1999.

[10] T. Solla and O. Vainio, "Comparison of programmable FIR filter architectures for low power," in Proc. 28th Eur. Solid-State Circuits Conf., Firenze, Italy, Sep. 2002, pp. 759–762.

[11] X. Chenghuan, C. He, Z. Shunan, and W. Hua, "Design and implementation of a high-speed programmable polyphase FIR filter," in Proc. 5th Int. Conf. Applicat.-Specific Integr. Circuit, vol. 2. Oct. 2003,

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

AUTHOR'S PROFILE

Mis. M. Anuvidhya
M.E - Applied Electronics in Magna College of Engineering under Anna University Chennai. TamilNadu India.
Email: anuvidhya04@gmail.com

Mr. A. M. S. Alaksander Jesus Gene M.E.
Senior Lecturer in the Department of Electronics & Communication,Magna College of Engineering Under Anna University Chennai Tamil Nadu India.
Email: alaksanderjesusgene@gmail.com