Realization of Distributed Arithmetic (DA) based reconfigurable digital FIR filter

Mayur B. Kachare[1], Prof. D. U. Adokar[2]
E&TC Department[1, 2], ME Scholar[1], Associate Professor[2], S.S.B.Ts C.O.E.T. Jalgaon
Email: mayur.kachare@rediffmail.com[1], dadokar@gmail.com[2]

Abstract- The Finite Impulse Response (FIR) filter is a digital filter widely used in digital signal processing applications in various fields like imaging, instrumentation, communications, etc. Programmable Digital Signal Processors (PDSPs) can be used in implementing the FIR filter. However, in realizing a large-order filter many complex computations are needed which affects the performance of the common digital signal processors in terms of speed, cost, flexibility, etc. Field-Programmable Gate Array (FPGA) has become an extremely cost-effective means of off-loading computationally intensive digital signal processing algorithms to improve overall system performance. The FIR filter implementation in FPGA, utilizing the dedicated hardware resources can effectively achieve Application Specific Integrated Circuit (ASIC) like performance while reducing development time cost and risks. In this paper, a band stop FIR filter is implemented on FPGA. Direct-form approach in realizing a digital filter is considered. This approach gives a better performance than the common filter structures in terms of speed of operation, cost, and power consumption in real-time. The FIR filter is implemented in FPGA and simulated with the help of Xilinx and MATLAB.

Index Terms- Distributed Arithmetic (DA), Finite Impulse Response (FIR), Least Mean Square (LMS), Look-Up Table (LUT), Multiply-Accumulate (MAC).

1. INTRODUCTION

A reconfigurable finite impulse response (FIR) filter's coefficients are changes during runtime have an important role in the software defined radio systems [2], multichannel filters [3], and digital up/down converters [4]. However, the famous multiple constant multiplications based technique [6], which is widely used for the implementation of FIR filters, cannot be used when the filter coefficients dynamically change. On the other hand, a general multiplier based design requires a large chip area and similarly enforces a limitation on the maximum possible order of the filter which is designed for high throughput applications. A distributed arithmetic (DA) based technique [6] has gained considerable importance and popularity in recent years for its high throughput processing capability and increased regularity, which results in profitable and area saving also time efficient computing structures. The main operations required for DA based computation are a sequence of lookup table (LUT) followed by shift register accumulation operations of the LUT output. The conventional DA used for design of an FIR filter assumes that impulse response coefficients are fixed, and this behavior tends to use ROM-based LUTs. The DA based implementation of FIR filters memory requirement exponentially increases with the filter order. To eliminate the problem of such a large memory requirement, systolic decomposition techniques are used for DA based implementation of long length convolutions and FIR filter of large orders [7, 8].

2. BACKGROUND

Distributed Arithmetic was firstly studied by Croisier [8] in 1973, and focused by Liu and Peled [9]. Quantization effects in DA system were studied by Kammeyer [10] and Taylor [11]. DA is used to design bit-level architecture for vector multiplications [2]. Traditionally, the input samples are used as addresses to access a series of LUTs whose entries are sums of coefficients. For filters implemented using DA, consider a discrete n-order fir filter with constant coefficients, also input samples coded as b-bit two's complement numbers with only the sign bit to the left of the binary point

$$x(n-k) = -x_{k0} + \sum_{j=1}^{B-1} x_{kj} 2^{-j} \qquad \dots (1)$$

Using Eq. (1) to compute the FIR output gives

$$y(n) = -\sum_{k=0}^{N} w_k x_{k0} + \sum_{j=1}^{B-1}\left[\sum_{k=0}^{N} w_k x_{kj}\right] 2^{-j} \qquad \dots (2)$$

These values can be pre-computed and stored in a LUT with the input used as the address. This technique provides the FIR filter having known coefficients to be implemented without general purpose multipliers. In this implementation a LUT requires exponentially increasing size with the number of taps N +1. Hence, reducing the LUT size improves area cost, as well as system performance.

2.1 Serial Distributed Arithmetic FIR Filter

A simplified view of a DA FIR is shown in Figure 1. In its most obvious and direct form, in serial distributed arithmetic (SDA) FIR DA based computations are bit serial in response. Extensions to the basic algorithm remove this limitation of promised throughput [2]. The advantage of a distributed arithmetic approach is its efficiency of mechanization. A sequence of additions, subtractions, look-ups tables and shifting of the input data sequence are the basic operations required. On FPGAs all of these functions precisely mapped. Input samples are applied to the input of parallel-to-serial shift register (PISO) at the input signal sample rate. When the new sample is converted to serial then bit wide output is provided to a time-skew buffer (TSB) or a bit-serial shift register. This buffer stores the input sample history in a bit-serial format and the required inner-product computation uses TSB for calculation. The TSB is itself constructed using a cascade connection of smaller bit–serial shift registers. The address inputs to a look-up table are nothing but the nodes in the cascade connection of TSBs. All possible products are stored [2] over the filter coefficient space by this LUT. Several observations provide valuable information of a DA FIR filter’s operation. The sample throughput is coupled to the filter length in a conventional multiply accumulate based FIR realization.

In DA architecture, the bit precision of the input data samples is related to the system sample rate. Every bit of an input sample has to be indexed and processed in turn before a new output sample is ready. Take *N*-bit precision input samples, *N*+1 clock cycles are required to form a new output sample for a symmetrical filter, and *N* clock cycles are needed for a non-symmetrical filter. The bit clock rate is the rate at which data bits are indexed. The bit frequency is greater than the filter sample rate (*fs*) and is equal to N**fs* for a non-symmetrical filter and (N+1)**fs* for a symmetrical filter. The required number of multiply and accumulate operations are taken using a time-shared or scheduled MAC unit in a conventional instruction set processor approach to solve a problem. The number of filter taps is inversely proportional to the filter sample throughput [12]. As the system sample rate is decreased, the filter length is proportionately increased or vice versa. This is not the case with DA based architectures [15]. As the filter length is increased in a DA FIR filter, large logic resources are used, but throughput is maintained.

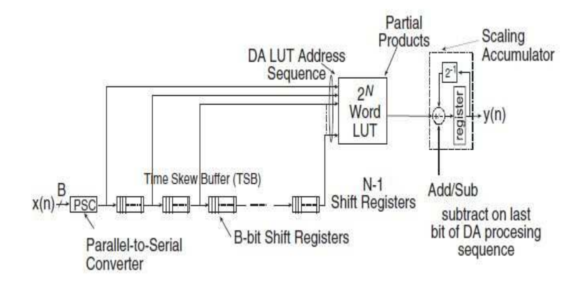


Fig.1. Serial distributed arithmetic FIR filter [15]

The filter sample rate is separated from the filter length. Here one of silicon area FPGA logic resources trade off taken place for time. As the filter length is increased in a DA FIR filter, throughput is maintained, but more logic resources are consumed.

2.2 Parallel Distributed Arithmetic FIR Filter

In its most obvious and direct form, DA based computations are bit serial in nature; every bit of the samples should have to be indexed before a new output sample becomes ready in SDA FIR. For a non symmetrical impulse response; When the input samples are represented with N bits of precision, N clock cycles are required to complete an inner-product calculation [12]. Additional speed can be obtained in lots of ways. Let one approach is to form parts of input words into M small words and process these small words parallel instead serially. This method requires M times as many memory lookup tables and hence increased storage are required of high cost. Maximum speed is achieved by factoring the input variables into single bit small words. This results in structure which is a fully parallel DA (PDA) FIR filter. With this factoring every new output sample is generated on every clock cycle. Such PDA FIR filters provide exceptionally high performance.

The FIR filter core is supported with implementations like parallel DA FIR which processes several bits in a clock period such filters may be designed where a totally parallel design that processes all the bits of the input data during a single clock period. If one considers a 12-bit precision input sample non symmetrical filter [16] and uses a serial

DA filter then new output samples are available every 12 clock periods. When data samples are processed 2 bits at a time (2-BAAT) then a new output sample is ready every 12/2 = 6 clock cycles. Also for 3-, 4-, 6- and 12-BAAT designs result is ready at every 4, 3, 2 and 1 clock cycles, respectively [13].

3. SYSTEM DESIGN

There has been remarkable progress in recent software tool development to support DSP applications in FPGAs kits. System Generator is a design tool for Xilinx FPGAs which extends the capabilities of Simulink in high-level to include bit and cycle accurate modelling of FPGA circuits, and generation of an FPGA circuit from a Simulink model [5, 6]. System Generator provides stronger Simulink libraries for memories, arithmetic and logic functions, and DSP functions. Also by supporting high level modelling and automatic code generation, System Generator creates new opportunities to determine the interplay between hardware-centric considerations and mathematical abstraction.

The entire design as realized in Simulink is shown in Figure 2.

3.1 System Generator:

The System Generator block has control of system and simulation parameters, and is used to assist the code generator. Every Simulink model having any element from the Xilinx Blockset must has at least one System Generator block. When a System Generator block is added to a model, it is easy and possible to find how code generation and simulation should be done.

3.2 FDATOOL:

The Xilinx FDATool block is MATLAB signal processing toolbox and it provides an interface to the FDATool software available. If the signal processing toolbox is not there then this block does not function properly and is of no use. This block gives a function of defining a FDATool object and stores it as a data of a System Generator model. FDATool gives a graphical user interface to digital filters. FDATool is very important tool in filter design.

3.3 Gateway IN:

The inputs to the Xilinx section of the Simulink design are the Xilinx Gateway In blocks. These blocks convert Simulink double, fixed-point and integer data types into the System Generator fixed-point type. The HDL design generated by System Generator is defined by every block of a top-level input port in.

3.4 Gateway OUT:

The outputs from the Xilinx section of the Simulink design are the Xilinx Gateway Out blocks. This block converts the System Generator fixed-point data type into Simulink Double.

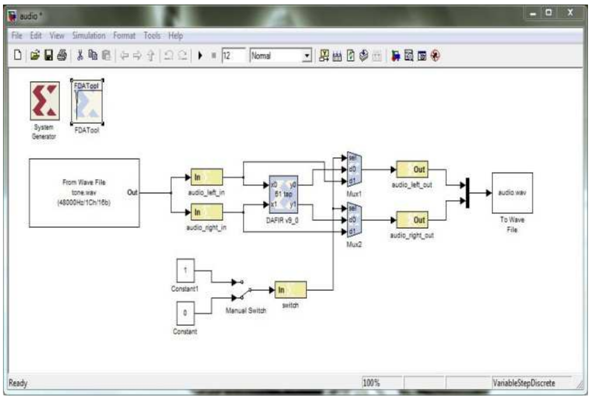


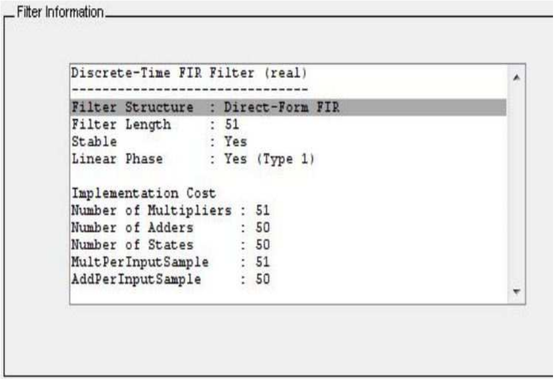
Fig.2. Designed distributed arithmetic FIR filter

3.5 Distributed Arithmetic FIR Filter v9.0 [15]:

- Highly parameterizable drop-in module for Virtex™, Virtex-E, Virtex-II, Virtex-II Pro, Virtex-4, Spartan™-II, Spartan-IIE, Spartan-3, and Spartan-3E FPGAs
- High-performance finite impulse response (FIR), half-band, Hilbert transform, interpolated filters, polyphase decimator, polyphase interpolator, half-band decimator and half-band interpolator implementations
- 2 to 1024 taps
- 1- to 32-bit input data precision
- Signed or unsigned input data
- Signed or unsigned filter coefficients
- 1- to 32-bit coefficient precision
- 1 to 8 channels
- Support for interpolation and decimation factors of between 1 and 8 inclusive Coefficient symmetry exploited (symmetric/negative-symmetric) to produce compact implementations
- Serial and parallel filters supported. The user may specify the degree of parallelism and trade off FPGA logic resources for sample rate in order to generate an optimal design
- Data-flow-style core interface and control
- On-line coefficient reload capability
- Incorporates Xilinx Smart-IP™ technology for maximum performance
- To be used with v7.1i or later of the Xilinx CORE Generator™ system

3.6 Multiplexer:

The Xilinx Mux block implements a multiplexer. The block has one select input (type unsigned) and a user-configurable number of data bus inputs, ranging from 2 to 1024.



```
Discrete-Time FIR Filter (real)
-------------------------------
Filter Structure  : Direct-Form FIR
Filter Length     : 51
Stable            : Yes
Linear Phase      : Yes (Type 1)

Implementation Cost
Number of Multipliers : 51
Number of Adders      : 50
Number of States      : 50
MultPerInputSample    : 51
AddPerInputSample     : 50
```

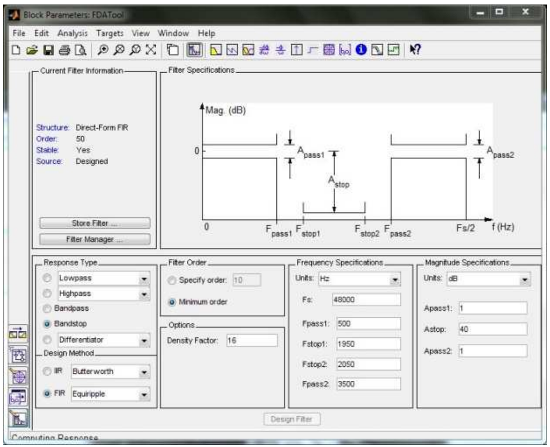
Fig.3. Designed FIR Filter information

3.7 MATLAB Simulink:

The design options in MATLAB allow the user to either create a code for designing filters that calls built-in functions, or to design filters in Sptool, a graphical user interface. MATLAB provides all the information necessary for building a hardware replica of the filter designed in software. Simulink is a data flow graphical programming language tool for modeling, simulating and analyzing multi-domain dynamic systems. Its primary interface is a graphical block diagramming tool and a customizable set of block libraries. It offers tight integration with the rest of MATLAB environment and can either drive MATLAB or be scripted from it. Simulink is widely used in control theory and digital signal processing for multi-domain simulation and model-based design. Filter Information is as shown below in figure 3.

4. RESULTS

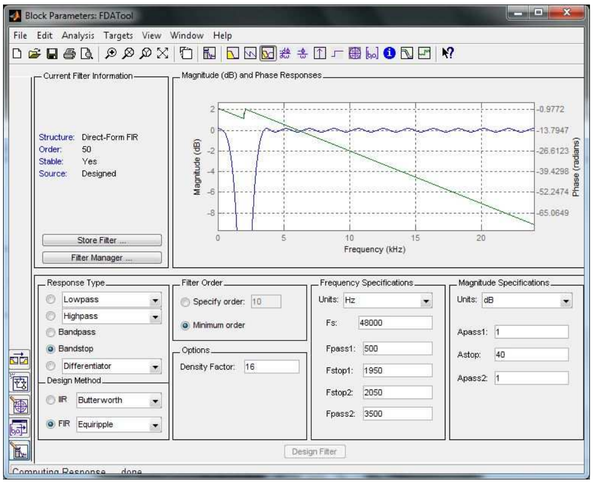
The specifications of a 50 order FIR filter in which attenuation obtain is below 40dB are shown in Fig. 4.



Structure: Direct-Form FIR
Order: 50
Stable: Yes
Source: Designed

Response Type: ○ Lowpass ○ Highpass ○ Bandpass ● Bandstop ○ Differentiator
Design Method: ○ IIR Butterworth ● FIR Equiripple
Filter Order: ○ Specify order: 10 ● Minimum order
Options: Density Factor: 16
Frequency Specifications: Units: Hz; Fs: 48000; Fpass1: 500; Fstop1: 1950; Fstop2: 2050; Fpass2: 3500
Magnitude Specifications: Units: dB; Apass1: 1; Astop: 40; Apass2: 1

Fig.4. Designed FIR Filter specifications



Structure: Direct-Form FIR
Order: 50
Stable: Yes
Source: Designed

Response Type: ○ Lowpass ○ Highpass ○ Bandpass ● Bandstop ○ Differentiator
Design Method: ○ IIR Butterworth ● FIR Equiripple
Filter Order: ○ Specify order: 10 ● Minimum order
Options: Density Factor: 16
Frequency Specifications: Units: Hz; Fs: 48000; Fpass1: 500; Fstop1: 1950; Fstop2: 2050; Fpass2: 3500
Magnitude Specifications: Units: dB; Apass1: 1; Astop: 40; Apass2: 1

Fig.5. Magnitude and Phase response of designed FIR Filter

The magnitude (dB) and phase response are shown in combined view in figure 5.

The comparison of given input audio and Output audio through FIR filter is shown in Fig. 6. Where the input audio is *.wav* file is a combination of signal with a noise of 2 KHz signal. Where the output audio is the noise free *.wav* signal.

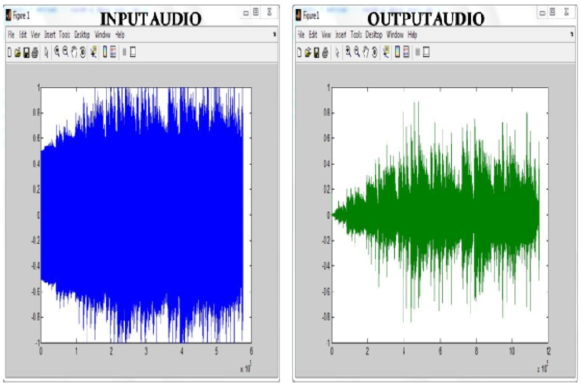


Fig.6. Comparison of Input audio & Output audio

5. CONCLUSION

This FIR filter is implemented using Matlab Simulink model and Xilinx System Generator for selected audio application. The input audio signal of which is noisy signal and it is filtered using FIR compiler block which is designed using FDA tool according to the sampling rate, passband frequency, stopband frequency, attenuations, transition width and filter orders and finally we got noiseless signal. The proposed FIR filter structure is verified using Xilinx system generator model and their performances are verified in terms of noise cancellation.

REFERENCES

[1] Sang Yoon Park and Pramod Kumar Meher, "Efficient FPGA and ASIC Realizations of DA-Based Reconfigurable FIR Digital Filter," IEEE Transactions on Circuits and Systems-II: Express Briefs, vol. 61, no. 7, pp. 511-515, July 2014.

[2] T. Hentschel, M. Henker, and G.Fettweis, "The digital front-end of software radio terminals," IEEE Pers. Commun. Mag., vol. 6, no. 4, pp. 40–46, Aug. 1999.

[3] K.-H. Chen and T.D. Chiueh, "A low-power digit-based reconfigurable FIR filter," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 53, no. 8, pp. 617–621, Aug. 2006.

[4] L. Ming and Y. Chao,"The multiplexed structure of multi-channel FIR filter and its resources evaluation," in Proc. Int. Conf. CDCIEM, Mar. 2012, pp. 764–768.

[5] I. Hatai, I. Chakrabarti, and S. Banerjee, "Reconfigurable architecture of a RRC FIR interpolator for multi-standard digital up converter," in Proc. IEEE 27th IPDPSW, May 2013, pp. 247–251.

[6] A. G. Dempster and M. D. Macleod, "Use of minimum-adder multiplier blocks in FIR digital filters," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., vol. 42, no. 9, pp. 569–577, Sep. 1995.

[7] S. A. White, "Applications of distributed arithmetic to digital signal processing: A tutorial review," IEEE ASSP Mag., vol. 6, no. 3, pp. 4–19, Jul. 1989.

[8] P. K. Meher, "Hardware-efficient systolization of DA-based calculation of finite digital convolution," IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 53, no. 8, pp. 707–711, Aug. 2006.

[9] P. K. Meher, S. Chandrasekaran, and A. Amira, "FPGA realization of FIR filters by efficient and flexible systolization using distributed arithmetic," IEEE Trans. Signal Process., vol. 56, no. 7, pp. 3009–3017, Jul. 2008.

[10] M. Kumm, K. Moller, and P. Zipf, "Dynamically reconfigurable FIR filter architectures with fast reconfiguration," in Proc. 8th Int. Workshop ReCoSoC, Jul. 2013, pp. 1–8.

[11] E. Ozalevli, W. Huang, P. E. Hasler, and D. V. Anderson, "A reconfigurable mixed-signal VLSI implementation of distributed arithmetic used for finite-impulse response filtering," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 55, no. 2, pp. 510–521, Mar. 2008.

[12] D. J. Allred, H. Yoo, V. Krishnan, W. Huang, and D. V. Anderson, "LMS adaptive filters using distributed arithmetic for high throughput," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 52, no. 7, pp. 1327–1337, Jul. 2005.

[13] P. K. Meher and S. Y. Park, "High-throughput pipelined realization of adaptive FIR filter based on distributed arithmetic," in Proc. IEEE/IFIP 19th Int. Conf. VLSI-SOC, Oct. 2011, pp. 428–433.

[14] DesignWare Building Block IP User Guide, Synposys, Inc., Mountain View, CA, USA, 2012, 06-SP2.

[15] LogiCORE DA FIR Filter v9.0, Xilinx, Inc., San Jose, CA, USA, 2005.

[16] S. Mandal, S.P. Ghoshal, R. Kar and D. Mandal, "Novel Particle Swarm Optimization for Low Pass FIR Filter Design", WSEAS transactions on signal processing, Issue 3, Volume 8, pp.111-120, Jul. 2012.

[17] C. Bauer, (2011) "Interactive Digital Signage – An Innovative Service and Its Future Strategies", Tirana, 2011 International Conference on Emerging Intelligent Data and Web Technologies (EIDWT), 7-9 September 2011, pp 137-142.

Author's Profile:

Mayur B. Kachare has recieved his B.E. graduation degree in Electronics and Communication in 2012 and now pursuing M.E. degree in Digital Electronics from SSBTs COET Bambhori, Jalgaon. He has participated in conference of special issue of IJECSCSE. He has published three papers in International Journals.

Dineshkumar U. Adokar has received his B.E. degree in Electronics from Visvesvaraya Regional College of Engineering, Nagpur in 1987 and Master's Degree in Electronics from Government College of Engineering, Amravati in 2001. He is also pursuing PhD from Sant Gadge Baba Amravati University. Presently, he is working as Associate professor in Department of Electronics and Telecommunication Engineering at SSBTs COET Bambhori, Jalgaon. He has published 10 research papers in national and international journals. His interests include Image processing, Microcontrollers etc.