International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 07 Issue: 08 | Aug 2020 www.irjet.net p-ISSN: 2395-0072

Comparative study of Digital FIR Filter (Using various Windows) on DSP Processor

Vaishali[1], Mr Mithlesh Kumar Singh[2]

[1]Student, M.Tech (ECE), Bhagwant Institute of Technology, Muzaffarnagar
[2]HOD ECE Department, Bhagwant Institute of Technology Muzaffarnagar

---

Abstract- *Digital filters are a very important part of DSP. In fact, their extraordinary performance is one of the key reasons that DSP has become so popular. A unique pipelined architecture for low-area, low-power, and high-throughput implementation of adaptive filter based on distributed using various Windows is presented in this paper. Distributed arithmetic (DA) is performed to design bit-level architectures for vector–vector Study involves Basics of Digital Filters discussed in Literature review. Architecture of TMS320C50 is discussed in chapter no. 2. MATLAB program is developed for calculating the filter coefficients. These are used in the assembly language program, which is implemented on TMS320C50 DSP Processor. Finally the comparison of features of above said windows is made based upon the obtained results. In future scope of the work, the adaptive filtering and its advantages are discussed. Also the finite word length effects and their remedies on FIR filter performance are discussed. The System is designed in Xilinx ISE 9.1 using Verilog HDL and it is routed using Model Sim 6.3. The Verification of the system's behavior is done using MATLAB 13. DA adaptive filters are advantageous over digital signal processing microprocessor in terms of total area and power consumption.*

1. INTRODUCTION

Most portable electronic devices, such as cellular phones, PDAs, and hearing aids, require digital signal processing (DSP) for high performance. Due to the increased demand of implementation of sophisticated DSP algorithms, lowcost designs, i.e. low-area and low-power cost, are needed to make these hand-held devices small with good performance. Generally, dedicated multipliers are expensive in terms of chip area and are frequently replaced by multiplier-free implementation methods. Distributed arithmetic (DA) stores the sums of scaled coefficients in a series of LUTs with binary inputs used as addresses [1][2]. Using DA, the MAC operations are replaced by adding the entries read from LUTs. In various applications, such as echo cancellation, time varying noise needs to be removed from desired signals. Hence, the coefficients of adaptive filters are updated based on the input samples[3][4].The adaptation of coefficients challenges the implementation of adaptive filters using DA, due to the high computation workload for updating the LUTs that store the sums of the scaled coefficients. This adaptation makes it challenging to implement DA-based adaptive filters with low cost due to the necessity of updating LUTs. One of the most famous algorithm which called least mean square (LMS) algorithm given by Widrow–Hoff is used here through which tapped-delay line finite impulse response(FIR) filter's weights are updated , These are simpler in design as well have satisfactory convergence performance [5].. Critical path due to inner-product computation to obtain filter output becomes very long so that for high sampling rate signals the critical path of design needed to be reduced to cope-up with the sampling period. Recently, high processing capability and regularity is achieved using the multiplier-less distributed arithmetic (DA) based technique [6] resulting in cost-effective and area–time efficient computing structures. Two separate lookup tables (LUTs) had been used by Allred et al. [7] which result in hardware-efficient DA-based design of adaptive filter. One lookup table for filtering and another for weight updation. Guo and DeBrunner [8], [9] have improved the design in [10] by using only one LUT for filtering as well as weight updating. However, the structures in [11, 12] do not support high sampling rate since they involve several cycles for LUT updates for each new sample. In a recent paper, we have proposed an efficient architecture for high-speed DA-based adaptive filter with very low adaptation delay [13]. Hence low power, low area, and high-throughput pipelined implementation of adaptive filter with very low adaptation delay is proposed here [14].

Thus this paper contributes to the following terms. 1) Throughput rate is significantly increased by a parallel LUT updation. 2) Further enhancement of throughput is achieved by concurrent implementation of filtering and weight updating. 3) Conventional adder-based shift accumulation is replaced by a conditional carry-save accumulation of signed partial inner products to reduce the sampling period. The bitcycle period amounts to memory access time plus 1-bit full-adder time (instead of ripple carry addition time) by carry-save accumulation. The area complexity of the design is reduced using carry-save accumulation method. 4) Reduction of power consumption is achieved by using a fast bit clock for carry-save accumulation but a much slower clock for all other operations.

2. MATERIAL & METHODS

FIR filters can be designed using various techniques. The most popular Window method is used for designing a FIR filter. FIR Filter is implemented using TMS320C50 Processor. Finally based upon the response of that filter, the response characteristics of filters based on various Windows like Rectangular, Hanning, Hamming and Blackman are compared.

2.1 Algorithms Used

2.2.1 For filter coefficient calculation using MATLAB:

1. Specify the Sampling Frequency and calculate the Nyquist frequency.
2. Specify the Pass band edge and Stop band edge frequencies. Calculate the Normalized Cutoff Frequency. Then calculate the Order of the filter.
3. Specify the Window function and calculate the Window coefficients. Then find rounded off filter coefficients in Q-15 format. (x 2^{15})
4. Convert them into Hexadecimal Equivalent and generate output in a text file, so that they can Be fired into the Program Memory of the DSP Processor.

2.2.2Assembly Program For implementation of FIR filter on TMS320C50 DSP Processor:

1. Move filter coefficients from Program memory to Data memory.
2. Take Input sample from ADC input port and store at appropriate memory location.
3. Set a counter based on the order of the filter N. Also clear accumulator to store the result.
4. Perform the linear Convolution between input samples x(n)(New sample & N-1 past samples) and N filter coefficients h(n) to generate output samples y(n).
5. Move the output sample at the DAC output port.
6. Repeat from step 2.

2.3 Window Method Functions supported in MATLAB.

2.3.1 Rectangular window

MATLAB Syntax w=**boxcar (**n)
w = **boxcar** (n,'*sfla++g*')
Description
w = boxcar (n) returns the n-point symmetric Rectangular window in the column vector w, where n is a positive integer.
w = boxcar (n,'*sflag*') returns an n-point Rectangular window using the window sampling specified by '*sflag*', which can be either 'periodic' or 'symmetric' (the default). When 'periodic' is specified, Rectangular computes a length n+1 window and returns the first n points.

2.3.2 Blackman window

MATLAB Syntax w = **blackman**(n)
w = **blackman**(n,'*sflag*')
Description
w = blackman(n) returns the n-point symmetric Blackman window in the column vector w, where n is a positive integer.
w = blackman(n,'*sflag*') returns an n-point Blackman window using the window sampling specified by '*sflag*', which can be either 'periodic' or 'symmetric' (the default). When 'periodic' is specified, blackman computes a length n+1 window and returns the first n points.
The equation for computing the coefficients of a Blackman window is

$$w[k+1] = 0.42 - 0.5\cos\left(2\pi\frac{k}{n-1}\right) + 0.08\cos\left(4\pi\frac{k}{n-1}\right), \quad k = 0, \ldots, n-1$$

Blackman windows have slightly wider central lobes and less sideband leakage than equivalent length Hamming and Hann windows.

2.3.3 Hamming window

MATLAB Syntax w = **hamming**(n)
w = **hamming**(n,'*sflag*')
Description
w = hamming (n) returns an n-point symmetric Hamming window in the column vector w. n should be a positive integer. The coefficients of a Hamming window are computed from the following equation.

$$w[k+1] = 0.54 - 0.46\cos\left(2\pi\frac{k}{n-1}\right), \qquad k = 0, \ldots, n-1$$

w = hamming(n,'*sflag*') returns an n-point Hamming window using the window sampling specified by '*sflag*', which can be either 'periodic' or 'symmetric' (the default). When 'periodic' is specified, hamming computes a length n+1 window and returns the first n points.

2.3.4 Hanning window

MATLAB Syntax w = **hann**(n)
w = **hann**(n,'*sflag*')
Description
w = hann(n) returns an n-point symmetric Hann window in the column vector w. n must be a positive integer. The coefficients of a Hann window are computed from the following equation.

$$w[k+1] = 0.5\left(1 - \cos\left(2\pi\frac{k}{n-1}\right)\right), \qquad k = 0, \ldots, n-1$$

w = hann(n,'*sflag*') returns an n-point Hann window using the window sampling specified by '*sflag*', which can be either 'periodic' or 'symmetric' (the default). When 'periodic' is specified, hann computes a length n+1 window and returns the first n points.

3. SYSTEM & IMPLEMENTATION

The ’C5x uses an advanced, modified Harvard-type architecture based on the ’C25 architecture and maximizes processing power with separate buses for program memory and data memory. The instruction set supports data transfers between the two memory spaces. Figure 2.13 shows a functional block diagram of the ’C5x. and figure 2.14 shows Block diagram of ’C5x.

The ’C5x architecture is built around four major buses:

- Program bus (PB)
- Program address bus (PAB)
- Data read bus (DB)
- Data read address bus (DAB)

The PAB provides addresses to program memory space for both reads and writes. The PB also carries the instruction code and immediate operands from program memory space to the CPU. The DB interconnects various elements of the CPU to data memory space. The program and data buses can work together to transfer data from on-chip data memory and internal or external program memory to the multiplier for single-cycle multiply/accumulate operations. The ’C5x has 96 registers mapped into page 0 of the data memory space. All ’C5x DSPs have 28 CPU registers and 16 input/output (I/O) port registers but have different numbers of peripheral and reserved registers. Since the memory-mapped registers are a component of the data memory space, they can be written to and read from in the same way as any other data memory location. The memory-mapped registers are used for indirect data address pointers, temporary storage, CPU status and control, or integer arithmetic processing through the ARAU.

4. RESULTS & DISCUSSION

By analysing the generated outputs, the inferences are made about the important features of the FIR filters based upon various Windows. Each Window has special characteristics like main lobe width, maximum side lobe magnitude and effect of frequency on

side lobe magnitude. Any window can be selected for the design of FIR filter, depending upon the desirable characteristics of that window.

Fig 4.1 shows the Rectangular window The width of main lobe in window spectrum is $4\pi/N$.The maximum side lobe magnitude in window spectrum is –13dB. In window spectrum the side lobe magnitude slightly decreases with increasing ω. In FIR filter design using rectangular window the minimum stop band attenuation is 22dB.

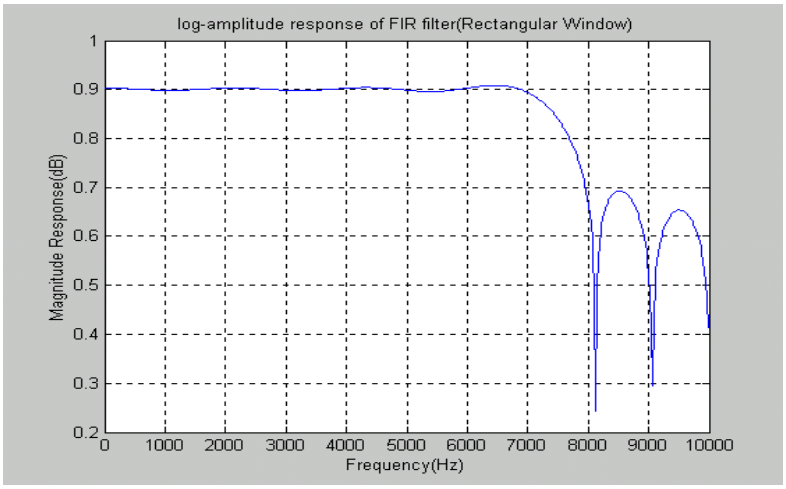

Figure 4.1 Log Amplitude Response of Rectangular Window

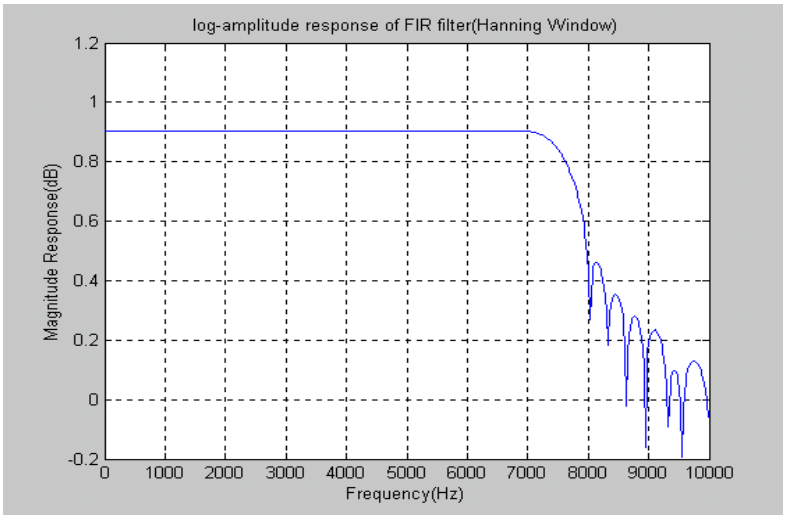

Figure 4.2 Log Amplitude Response of Hanning Window

Fig 4.2 shows Hanning The width of main lobe in window spectrum is $8\pi/N$. The maximum side lobe magnitude in window spectrum is –31dB. In window spectrum, the side lobe magnitude decreases with increasing ω. Using Hanning window, the minimum stop band attenuation is 44dB.

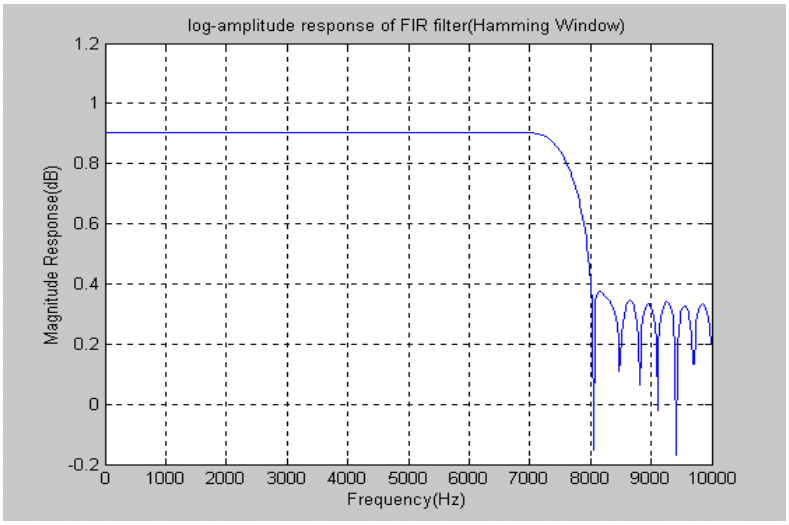

Fig 4.3 Log Amplitude Response of Hamming Window

Fig 4.3 shows hanning windows the width of main lobe in window spectrum is $8\pi/N$. The maximum side lobe magnitude in window spectrum is –41dB.In window spectrum, the side lobe magnitude remains constant. In FIR filter design using Hamming window, the minimum stop band attenuation is 51dB.

Fig 4.4 shows the analysis of Blackman windows. The width of main lobe in window spectrum is $12\pi/N$. The maximum side lobe magnitude in window spectrum is –58dB. In window spectrum, the side lobe magnitude rapidly decreases with increasing ω. In FIR filter design using Blackman window, the minimum stop band attenuation is 78dB.

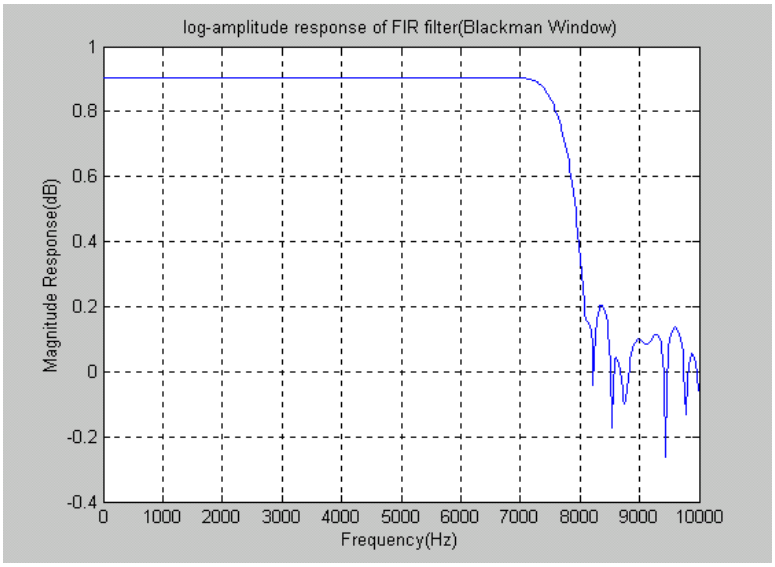

Figure. 4.4 Log Amplitude Response of Blackman Window

5. CONCLUSIONS

There are four ways in which these performance of FIR filter is improved.

a.) ADC Noise: caused by limited no. of bits to be used for ADC output, which results in lower S/N ratio. The error due to this can be avoided by using additional bits for ADC output

b.) Coefficient Quantization errors: this result from representing filter coefficients with a limited no of bits. This can be solved using enough bits to represent filter coefficients. Different optimization techniques can be used for efficient selection of the coefficients to minimize these errors.

c.) Round off Errors from Quantizing results of arithmetic operations: These occur when the lower order bits are discarded before storing the results of multiplications. This is normally forced due to word length limitations of the processors used. Rounding after double length summing of products may reduce the error due to this effect.

d.) Arithmetic Overflow: This occurs when partial sums or filter output exceeds the word length of the processor. This results in wrong output samples (normally the sign changes). To avoid this overflow the filter coefficients can be scaled by dividing each filter coefficient by a factor such that the output sample never exceeds the permissible world length.

The availability of powerful DSP hardware and software tools, have made it feasible to realize the complex filtering operations. By applying the low power consumption techniques like Booths algorithms, the power consumption in portable DSP based digital devices like mobile phones can be reduced. Adaptive filtering advantage of achieving efficient filtering at continuously changing conditions and the effect of Fine word length effects and their remedies are discussed in the future scope of the work.

REFERENCES

1. S. A. White, “Applications of distributed arithmetic to digital signal processing::A tutorial review,” IEEE ASSP Mag., vol. 6, pp. 4–19, Jul. 1989.
2. B. Farhang-Boroujeny, Adaptive Filters: Theory and Applications. Chichester, U.K.: Wiley, 1998.S. Haykin, Adaptive Filter Theory. Upper Saddle River, NJ: Prentice-Hall, 1996.
3. 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.
4. S. Haykin and B. Widrow, Least-Mean-Square Adaptive Filters. Hoboken, NJ, USA: Wiley, 2003.
5. D. J. Allred, H. Yoo, V. Krishnan, W. Huang, and D. V. Anderson, “An FPGA implementation for a high throughput adaptive filter using distributed arithmetic,” in Proc. 12th Annu. IEEE Symp. Field-Programmable Custom Comput. Mach., 2004, pp. 324– 325.
6. R. Guo and L. S. DeBrunner, “Two high performance adaptive filter implementation schemes using distributed arithmetic,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 58, no. 9, pp. 600–604, Sep. 2011.
7. R. Guo and L. S. DeBrunner, “A novel adaptive filter implementation scheme using distributed arithmetic,” in Proc. Asilomar Conf. Signals, Syst., Comput., Nov. 2011, pp. 160–164.
8. P. K. Meher and S. Y. Park, “High-throughput pipelined realization of adaptive FIR filter based on distributed arithmetic,” in VLSI Symp. Tech. Dig., Oct. 2011, pp. 428–433.
9. A. Peled and B. Lie, “A new hardware realization of digital filters,”IEEE Trans. Acoustics, Sound, Signal Process., vol. ASSP-22, no. 4, pp.456–462, Dec. 1974.
10. Croisier, D. J. Esteban, M. E. Levilion, and V. Rizo, “Digital Filter for PCM Encoded Signals,” U.S. Patent 3 777 130, Apr. 1973.
11. D.J. Allred,V. Krishnan,W. Huang, and D. Anderson, “Implementation of an LMS adaptive filter on an FPGA employing multiplexed multiplier architecture,” in Proc. Asilomar Conf. Signals, Systems, Computers., Nov. 2003, pp. 918–921.
12. Kar R, Mandal D, Mondal S, Ghoshal SP. Craziness based particle swarm optimization algorithm for FIR band stop filter design. Swarm Evol Comput 2012;7:58–64.
13. Kaur R, Patterh MS, Dhillon JS, Singh D. Heuristic search method for digital IIR filter design. WSEAS Trans Signal Process 2012;8(3):121–34.
14. Vasundhara, Mandal D, Ghoshal SP, Kar R. Digital FIR filter design using hybrid random particle swarm optimization with differential evolution. Int J Comput Intell Syst 2013;6(5):911–27.
15. Karboga N, Cetinkaya MB. A novel and efficient algorithm for adaptive filtering: artificial bee colony algorithm. Turk J Electr Eng Comput Sci 2011;19(1):175–90.