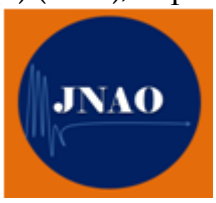
Journal of Nonlinear Analysis and Optimization
Vol. 11(9) (2020), September 2020
https://ph03.tci-thaijjo.org/
ISSN : 1906-9685



J.Nonlinear Anal. Optim

DESIGN OF MODIFIED DA CONCEPT ON FPGA FOR FIR FILTER

Dr. K.Venkataramana
Associate Professor, Department of CSE
Sri Sai Institute of Technology and Science, Rayachoti
Email: venkataramanaphd@gmail.com

Abstract –The implementation of FIR filters on FPGA based on traditional method costs considerable hardware resources, which goes against the decrease of circuit scale and the increase of system speed. A new design and implementation of FIR filters using Distributed Arithmetic is provided in this paper to solve this problem. Distributed Arithmetic structure is used to increase the resources usage while pipeline structure is also used to increase the system speed. In addition, the devided LUT method is also used to decrease the required memory units. The simulation results indicate that FIR filters using Distributed Arithmetic can work stable with high speed and can save almost 50 percent hardware resources to decrease the circuit scale, and can be applied to a variety of areas for its great flexibility and high reliability.

Keywords –Distributed Arithmetic; FIR; pipeline; LUT; FPGA

I. INTRODUCTION

Digital filters are the essential units for digital signal processing systems. Traditionally, digital filters are achieved in Digital Signal Processor (DSP), but DSP-based solution cannot meet the high speed requirements in some applications for its sequential structure. Nowadays, Field Programmable Gate Array (FPGA) technology is widely used in digital signal processing area because FPGA-based solution can achieve high speed due to its parallel structure and configurable logic, which provides great flexibility and high reliability in the course of design and later maintenance. In general, Digital filters are divided into two categories, including Finite Impulse Response (FIR) and Infinite Impulse Response (IIR). And FIR filters are widely applied to a variety of digital signal processing areas for the virtues of
providing linear phase and system stability.

The FPGA-based FIR filters using traditional direct arithmetic costs considerable multiply-and accumulate (MAC) blocks with the augment of the filter order. However, according to Distributed
Arithmetic, we can make a Look-Up-Table(LUT) to conserve the MAC values and callout the values according to the input data if necessary. Therefore, LUT can be created to take the place of MAC units so as to save the hardware resources.

This paper provide the principles of Distributed Arithmetic, and introduce it into the FIR filters design, and then presents a 31-order FIR low-pass filter using Distributed Arithmetic, which save considerable MAC blocks to decrease the circuit scale, meanwhile, devided LUT metherd is used to decrease the required memory units and pipeline structure is also used to increase the system speed.

II. DISTRIBUTED ARITHMETIC

Distributed Arithmetic was first brought up by Croisier, and was extended to cover the signed data system by Liu , and then was introduced into FPGA design to save MAC blocks with the development of FPGA technology. Distributed Arithmetic was first brought up by Croisier.

The N-length FIR filter can be described as:

$$y =< h,x >= \sum_{n=0}^{N-1} h[n]x[n]$$

Where h[n] is the filter coefficient and x[n] is the input sequence to be processed. The FIR structure consists of a series of multiplication and addition units, and consumes N MAC blocks of FPGA, which are expensive in high speed system. Compared with traditional direct arithmetic, Distributed Arithmetic can save considerable hardware resources through using LUT to take the place of MAC units. Another virtue of this method is that it can avoid system speed decrease with the increase of the input data bit width or the filter coefficient bit width, which can occur in traditional direct method and consume considerable hardware resources.

Distributed Arithmetic is introduced into the design of FIR filters as follows. In the two's complement system, x[n] can be described as:

$$x[n] = -2^{B} x_{B}[n] + \sum_{b=0}^{B-1} 2^{b} x_{b}[n]$$

Substituting equ.(2) into equ.(1) yields:

$$y = -2^{B} x_{B}[n]h[n] + \sum_{b=0}^{B-1} h[n] \sum_{n=0}^{N-1} 2^{b} x_{b}[n]$$

The second part of the equ. Can be changed into another form:

$$\sum_{b=0}^{B-1} h[n] \sum_{n=0}^{N-1} 2^{b} x_{b}[n] = \sum_{b=0}^{B-1} 2^{b} \sum_{n=0}^{N-1} h[n]x_{b}[n]$$

Substituting equ. (4) Into equ. Yields to the final form of Distributed Arithmetic:

$$y = -2^{B} x_{B}[n]h[n] + \sum_{b=0}^{B-1} 2^{b} \sum_{n=0}^{N-1} h[n]x_{b}[n]$$

Take a close look at the right part of equ. Considering the limited possibility of input data, we can conserve the values of $\sum_{n=0}^{N-1} h[n]x_{b}[n]$ into a LUT unit and then callout the Revalent value according to the input data to save MAC blocks. And then the weighted sum of $\sum_{n=0}^{N-1} h[n]x_{b}[n]$ is calculated through shift registers the result is $\sum_{b=0}^{B-1} 2^{b} \sum_{n=0}^{N-1} h[n]x_{b}[n]$ In signed system, the signed bit should be taken into consideration so $-2^{B} x[n]h[n]$ is also added. As a result, the final form of Distributed Arithmetic is defined as equ. (5) and the implementation can be achieved on FPGA through LUT units. As the expatiation above, the basic Distributed Arithmetic structure can be described as Fig.1. The dotted rectangle is the register.

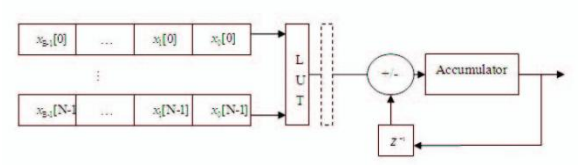


Fig.1 The basic Distributed Arithmetic structure

According to the input sequence, we can conserve the
Coefficient values in LUT unit, the LUT constructing formula is given in Tab.1.

Tab.1 LUT constructing formula

$x_b[N-1]$	$x_b[N-2]$	…	$x_b[0]$	Coefficient value
0	0	…	0	0
…	…	…	…	…
b_{N-1}	b_{N-2}	…	b_0	$h[0]\,b_0+\ldots+h[N-1]\,b_{N-1}$
…	…	…	…	…
1	1	…	1	$h[0]+\ldots+h[N-1]$

With above structure and coefficient values of LUT, we can achieve a variety of filters to meet various requirements.

III.FILTERS DESIGN

In the course of FIR filters design, ranging can be generated at the edge of transition band for the reason that finite series Fourier transform cannot produce sharp edges. So windows are often used to produce suitable transition band, and Kaiser Window is widely used for providing good performance. The parameter is an important coefficient of Kaiser Window which involves the windows types. We can get a variety of windows like Rectangular window, Hanning window, Hamming window, and Blackman window with the adjustment of β.

A 31-order FIR low-pass filter is designed using Kaiser Window, and the parameter is as follows: =3.39, w=0.18. We can obtain the filter coefficients using Mat lab as follows.
h(0)=h(31)=0.0019;h(1)=h(30)=0.0043;h(2)=h(29)=0.0062;h(3)=h(28)=0.0061;h(4)=h(27)=0.0025;h(5)=h(26)=-0.0050;h(6)=h(25)=-0.0148;h(7)=h(24)=-0.0236;h(8)=h(23)=0.0266;h(9)=h(22)=0.0192;h(10)=h(21)=0.0015;h(11)=h(20)=0.0351;h(12)=h(19)=0.0774;h(13)=h(18)=0.1208;h(14)=h(17)=0.1566;h(15)=h(16)=0.1768.

In Matlab data is described in the floating-point form while described in the fixed-point form in this FPGA system. After quantizing the filter coefficients using 12-bit-width signed binary, we can obtain the final coefficients as follows:
h(0)=h(31)=4;h(1)=h(30)=9;h(2)=h(29)=13;h(3)=h(28)=12;h(4)=h(27)=5;h(5)=h(26)=-10;h(6)=h(25)=-30;h(7)=h(24)=-48;h(8)=h(23)=-55;h(9)=h(22)=-39;h(10)=h(21)=3;h(11)=h(20)=72;h(12)=h(19)=158;h(13)=h(18)=247;h(14)=h(17)=321;h(15)=h(16)=362.With above coefficients in Matlab, the frequency-amplitude characteristic of the filter is described as Fig.2.

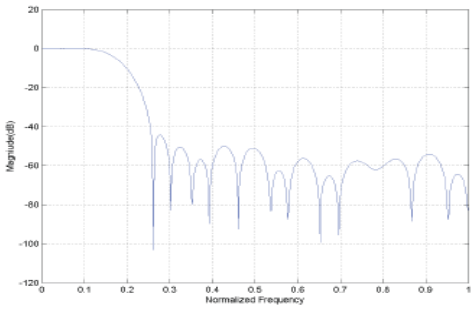


Fig.2 Frequency-amplitude characteristic of the filter

From Fig.2, we can observe that the low-pass filter has a good permanence for low-pass filtering, and the cut-off frequency is 2.88 MHz if the sampling frequency is defined as 32 Mhz. We can achieve the filter on FPGA

according to Fig.1 and Tab.1. However, considering the even symmetry of the coefficients, we can use equ. (6) to simplify the system . And the final values are definite as the input of shift registers.

$$y[i]=x[i]+ x[31-i] \quad i=0,1...15$$

However, with the increase of filter order, the scale of LUT will increase dramatically, which will cost more time to look up the table and more memory to store the values. Therefore, we can divide the LUT unit into four small LUT units to solve this problem. Then the value of the four devided LUT units is added as the final value. Coefficient values of small LUT is given in Tab.2.

Tab.2 Coefficient values of LUT

$b_3b_2b_1b_0$	Data
0000	0
0001	h[0]
0010	h[1]
0011	h[0]+ h[1]
0100	h[2]
0101	h[0]+ h[2]
0110	h[1]+ h[2]
0111	h[0] + h[1]+ h[2]
1000	h[3]
1001	h[0]+ h[3]
1010	h[1]+ h[3]
1011	h[0]+ h[1]+ h[3]
1100	h[2]+ h[3]
1101	h[0]+ h[2]+ h[3]
1110	h[1]+ h[2]+ h[3]

Pipeline structure is also used to increase the system speed. The pipelining technology is to divide combinational circuit into small parts, and then insert a register in the middle of the two parts to increase the system speed. The filter designed in this paper
Contains 3 level registers. Although it will increase the time delay, but helps to increase the system speed.
Considering all the factors above, we achieve the new structure based on Distributed Arithmetic as Fig.3.

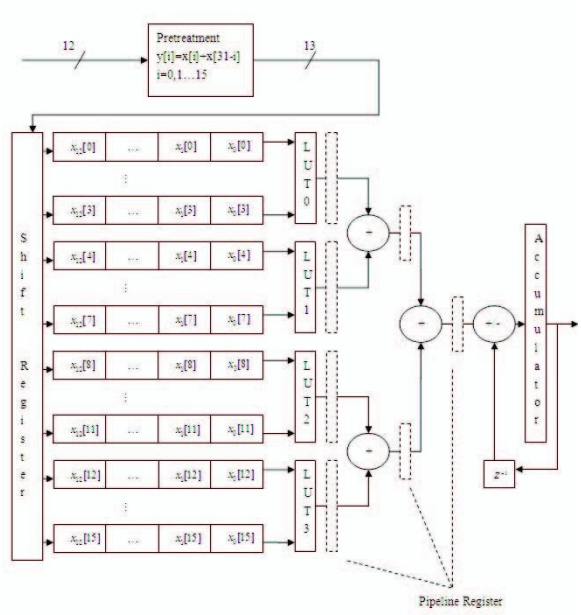


Fig.3 Structure of FIR filter based on Distributed Arithmetic

The input data is defined as 12-bit-width complement, and the system can also process signed signals. According to the structure above, we achieve the whole design using Verilog language in Quartus 6.2.

V. IMPLEMENTATION AND RESULTS

The proposed is Distributed Arithmetic designed using Verilog hard ware description language and structural form of coding. The basic block of design is luts. The design is completely synchronized by the clock..The code is completely synthesized using Xilinx XST and implemented on device family Virtex-5, device XC5VL50, package FF324 with speed grade -2.

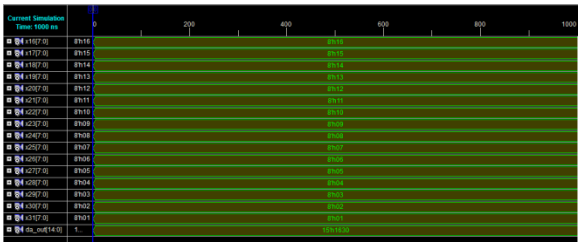


Fig 4 Simulation results

Mat lab and Modelsim are used as the simulation platforms. We can analysis the changes between the input wave and the output wave to observe the permanence of the designed filter through Mat lab , while observing the real-time implementation performance of FPGA through Modelsim.

To observe the performance of the designed filter, a square signal of 1.6MHz is generated as the input by Mat lab, and the sampling frequency is 32MHz. The data is changed into the two's complement form and stored in the file named fir_in.txt. And the data is called out later as the input of the FPGA filter through test bench language in Modelsim, the output data is shown in the waveform workstation and also stored into the file named fir_out.txt, which can be read by Mat lab to make a contrast to analysis.

In Fig.4, the waveforms are in frequency domain. They are the frequency spectrums of input signal, filter and output signal respectively. We can observe that the output wave only contains low frequency signal after the implementation of the filter. We can conclude that the filter coefficients are suitable for the test.

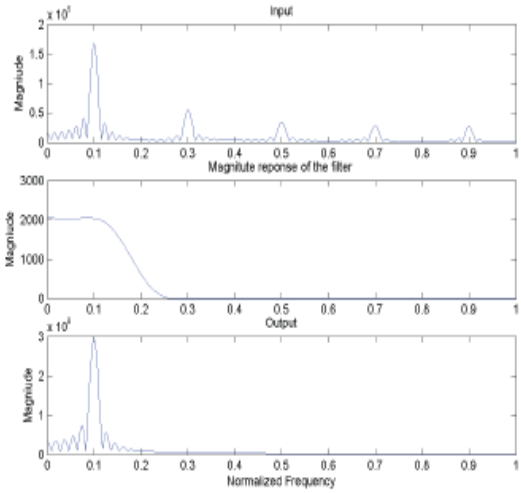


Fig.4 Frequency domain waveform

In Fig.5, the waveforms are in time domain. They are the input signal, output signal, and the filted signal by FPGA filter respectively. We can observe that output filted by FPGA filter is almost the same as the output simulated by Matlab, the permanence of the designed filter is perfect.

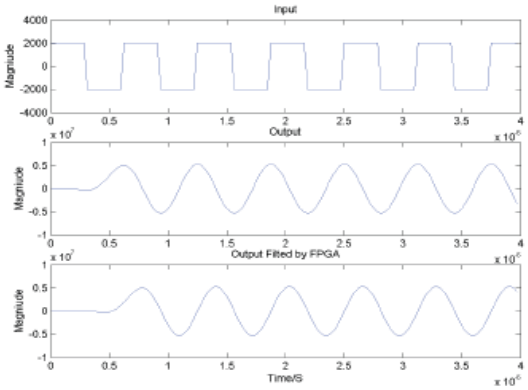
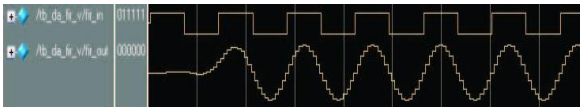


Fig.5 Time domain waveform

In Fig.6, it is in the Modelsim simulation environment, and the time delays is just the time of the implementation of FPGA filter.



The test results above show that the filter works stable, and completely meets the designed requirements. It cost 2221 logic elements and obtain 220M system speed using traditional direct arithmetic, while only cost 1110 logic elements and obtains 230M system speed using Distributed Arithmetic. Therefore, we can save almost 50 percentage of the hardware resources using Distributed Arithmetic to decrease the hardware scale. And we can obtain a higher speed if we sacrifice the hardware resources for speed using other relevant technologies.

V. CONCLUSION

This paper presents the design and implementation based on Distributed Arithmetic, which is used to realize a 31-order FIR low-pass filter. Distributed Arithmetic structure is used to increase the resources usage while pipeline

structure is used to increase the system speed. The test results indicate that the designed filter using Distributed Arithmetic can work stable with high speed and can save almost 50 percent hardware resources'. Meanwhile, it is very easy to transplant the filter to other applications through modifying the order parameter or bit width and other parameters, and therefore have great practical applications in digit signal processing.

REFERENCES

[1] Uwe Meyer-Baese.Digital signal processing with FPGA[M]. Beijing:Tsinghua University Press,2006:50~51

[2] Tsao Y C and Choi K. Area-Efficient Parallel FIR Digital Filter Structures for Symmetric Convolutions Based on Fast FIR Algorithm [J]. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 2010,PP(99):1~5.

[3] Chao Cheng and Keshab K Parhi. Low-Cost Parallel FIR Filter Structures With 2-Stage Parallelism[J].*IEEE Transactions on Circuits and Systems I*: *Regular* ,2007,54(2):280~290.

[4] Tearney G J and Bouma B E. Real-Time FPGA Processing for High-Speed Optical Frequency Domain Imaging [J]. *IEEE Transactions on Medical Imaging,* 2009,28(9):1468~1472.

[5] Hu Guang-shu. Digital signal processing-theory,algorithm and realizes[M]. 2nd ed.Beijing: Tsinghua University Press,2003:296~307.

[6] Chun Hok Ho,Chi Wai Yu and Leong P. Floating-Point FPGA: Architecture and Modeling [J]. *IEEE Transactions on Very Large Scale Integration Systems*, 2008,17(12): 1709~1718.