Journal of Basic and Applied Engineering Research
Print ISSN: 2350-0077; Online ISSN: 2350-0255; Volume 2, Number 10; April-June, 2015 pp. 822-825
© Krishi Sanskriti Publications
http://www.krishisanskriti.org/jbaer.html

An Effective Finite Impulse Response Filter Realization using Modified Distributed Arithmetic for High Speed Applications

Vaibhavi Nautiyal[1] and M. Vinoth Kumar[2]

[1]M.Tech – VLSI Design Department of ECE SRM University – NCR Campus
[2]Department of ECE SRM University – NCR Campus
E-mail: [1]vaibhavinautiyal1990@gmail.com, [2]mvkeie@gmail.com

Abstract—*The design of an Effective Finite Impulse Response (FIR) filter is done using Modified Distributed Arithmetic and Pipelining technology for an efficient implementation on FPGA to increase the performance of the filter in according to increase speed and reduce power dissipation. The FIR filters traditionally used become less efficient with increase in size. Distributed Arithmetic is used where the Look up Table structures replace the multiply and accumulate units conserving hardware resources and reducing area and system latency. This paper presents the modified distributed arithmetic architecture and pipelined structure which cause increment in the system speed. The divided LUT technique is used to reduce required memory usage. Modified Distributed Arithmetic FIR filter can work with stability at higher speeds while conserving 50 % area usage and providing flexibility to use the FIR filter in different areas of application.*

1. INTRODUCTION

The digital filters are an essential part of digital signal processing systems and are responsible for enhancing or reducing certain features of the sampled signals using mathematical operations. With the advancement in technology the FIR filter designing methods have also seen significant changes. Earlier the FIR filters were designed using multipliers and adders then the pipelining and parallel processing were introduced to increase speed and reduce delay. After the year 2000 the era of small size came in and the size reduction of the FIR filter became significant. This led to introduction of various new algorithms which can be implemented in the FIR filters to reduce complexity, power consumption and area. Multiplication is the main source of complexity in the FIR filters. The multiplication consumes most of the area and delays the throughput in addition to its high consumption of power [2]. Thus the multipliers put a limitation on the number of Processing Elements (PE) and the realization of higher order filters [3].The FIR filters are commonly used due to their linear phase property and simple implementation process. In this paper we discuss recent progress in FIR filter designing technology.

2. DRAWBACKS AND PROGRESS

In the last three decades the FIR filter designing has seen various changes. For achieving a perfect and efficient architecture the technologies like pipelining, parallel processing, Low power multiplication, common sub-expression elimination, genetic algorithm, distributed arithmetic, and DA LUT were proposed and implemented. Over the years the technologies used brought a distinct reduction in the amount of area used in the FIR implementation. The Fig. 1 [6], [5] describes reduction in area due to the different designing techniques used.

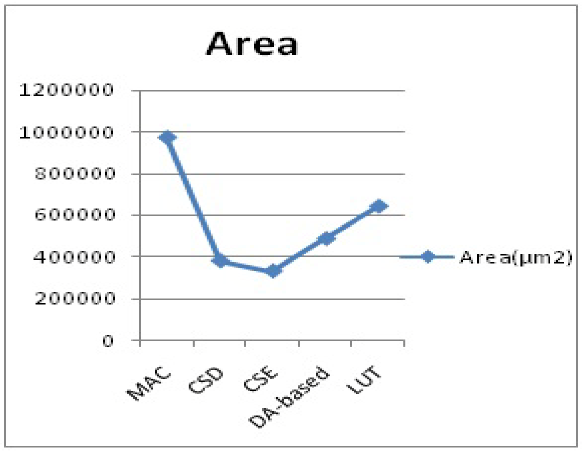

Fig. 1: Comparison of area of various FIR filter designing techniques used.

In the given Fig. 1 the MAC (multiply and accumulate),Add and shift scheme with CSD encoding, Common sub-expression elimination, Distributed Arithmetic and LUT(lookup table) partitioning methods are compared for area used in implementation of 40 tap FIR filter. The pipelining and parallel processing were used in the conventional method in order to overcome the latency issues in the FIR filter.

Pipelining involves the inserting latches in the system. Thus any operation along the critical path is broken into smaller and quicker operations with registers between the levels in order to increase the operating frequency which leads to higher throughput [2]. The other process used is parallel processing in which the multiple inputs are processed simultaneously to give multiple outputs [2]. Thus parallel processing and pipelining involves use of extra hardware. The table [7] shows the throughput and latency comparisons for different realizations of trapezoidal rule:

Table 1: Throughput and Latency comparisons for different realizations of Trapezoidal rule.

Structure	Throughput (MHz)	Latency (No. Of clock cycles)	Clock Period (ns)
Basic FIR	92.52	4	10.809
Transposed FIR	128.13	4	7.804
Pipelined FIR	189.5	5	5.277
Fine-grain pipelined FIR	387.45	6	2.581
4-parallel FIR	1354.52	6	2.581

The Table 1 compares the Latency and throughput of 4 tap FIR filter with input word length of four bits. The FIR filters of conventional method with increase in number of taps have drawbacks of delay, power consumption and large area. FIR filter consume more power and area due to the multiply and accumulate (MAC) units.

In Fig. 2 the power dissipation of different FIR filter realization is shown at nominal operating frequency [7].The implementation of FIR filter with less area and less latency has become essential thus memory based structures are used. These give regularity, reduced latency in implementation and have less dynamic power consumption due to less switching activities for memory read operations compared to conventional multipliers [4]. So the multiplier block is replaced by the Look up tables (LUT), shifter and accumulator using the Distributed Arithmetic Algorithm. The Look up table is single bit memory cells used to store the bit value [4].Thus the DA based FIR filters have provided a simple realization with reduced area and power consumption.

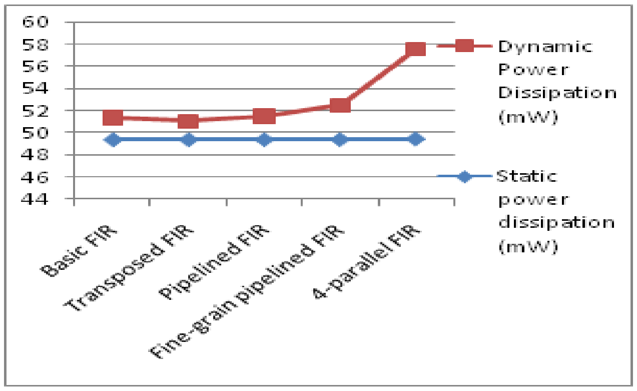

Fig. 2: Power dissipation for Different FIR realizations

3. DISTRIBUTED ARITHMETIC ARCHITECTURE IN FIR FILTER

With increase in the order of the filter the size of the filter increases and the complexity also increases. In order to provide simpler structure at higher orders with less latency several algorithms have been proposed. The Distributed Arithmetic is a computational algorithm which does efficient implementation of weighted sum of products or dot products [5].DA is bit serial operation used to compute the inner product of a constant coefficient vector and a variable input vector which is given by equation (1) is [5]:

$$y = \sum_{k=1}^{K} A_k b_k$$

y is output response

A_k is constant filter coefficient

X_k is input data

On rearranging and grouping the final formulation in the equation 2 is:

$$y = -\sum_{k=1}^{K} A_k . b_{k0} + \sum_{n=1}^{N-1} [\sum_{k=1}^{K} A_k . b_{kn}] 2^{-n}$$

3.1 DA Technique for 7th order FIR Filter

The DA technique for 7th order FIR filter using Matlab coefficients can be given as [5]:

No. of coefficients=8

No. of inputs=8

The calculated MATLAB coefficients are:

h0 =00000000

h1=00000011

h2=11100101

h3=01101001

h4=10010110

h5=11100101

h6=00000011

h7=00000000

The possible outputs are pre computed and stored in the LUT which are addressed by input of the filter. The possible input for 8 tap Fir filter is from 00000000 to 11111111.

- Input=00000000
- Output=0.h0+0.h1+0.h2+0.h3+0.h4
  +0.h5+0.h6+0.h7
  =0
- Input=00000001
- Output=0.h0+0.h1+0.h2+0.h3+0.h4

+0.h5+0.h6+1.h7
=h7

- Input=00011101
- Output=0.h0+0.h1+0.h2+1.h3+1.h4

  +1.h5+0.h6+1.h7
  =h3+h4+h5+h7
- Input=00000111
- Output=0.h0+0.h1+0.h2+0.h3+0.h4

  +1.h5+1.h6+1.h7
  =h5+h6+h7
- Input=00010000
- Output =0.h0+0.h1+0.h2+1.h3+0.h4

  +0.h5+0.h6+0.h7
  =h3
- Input=00000100
- Output =0.h0+0.h1+0.h2+0.h3+0.h4

  +1.h5+0.h6+0.h7
  =h5
- Input=01000011
- Output=0.h0+1.h1+0.h2+0.h3+0.h4
  +0.h5+0.h6+0.h7
  =h1
- Input=01000011
- Output=0.h0+1.h1+0.h2+0.h3+0.h4
  +0.h5+1.h6+1.h7
  =h1+h6+h7
- Input=00000010
- Output=0.h0+0.h1+0.h2+0.h3+0.h4
  +0.h5+1.h6+0.h7
  =h6

Let consider following calculations to illustrate the technique:

The inputs are X0, X1, X2, X3, X4, X5, X6 and X7.
X0=00000000=0

X1=00000001=1

X2=00011101=29

X3=00000111=7

X4=00010000=16

X5=00000100=4

X6=01000011=67

X7=00000010=2

Step 1
X0[0] X1[0]…X6[0] X7[0] =00000000

X0[1] X1[1]…X6[1] X7[1] =00000010

X0[2] X1[2]…X6[2] X7[2] =00000000

X0[3] X1[3]…X6[3] X7[3] =00101000

X0[4] X1[4]…X6[4] X7[4] =00100000

X0[5] X1[5]…X6[5] X7[5] =00110100

X0[6] X1[6]…X6[6] X7[6] =00010011

X0[7] X1[7]…X6[7] X7[7] =01110010

Step 2
Output from LUT

O1=00000000=0

O2=00000011=3

O3=00000000=0

O4=101001110=334

O5=11100101=229

O6=1001100000=608, O7=110000001=385

Step 3
Output=O1+One time shifted value of O2+two time shifted value of O3+Three time shifted value of O4+Four time shifted value of O5+Five time shifted value of O6+Six time shifted value of O7
Output =0+6+0+117+94+19+112
=348
The number of stored values in the look up table increases exponentially with the order of the filter. For 8 tap filter the number of locations is 256.Thus another concept of LUT partitioning is introduced.

4. LUT PARTITIONING

The LUT partitioning divides the LUT such that number of LUT reduces yet the values stored remain the same [3].The realization of 16 tap FIR filter with various partitions using the Matlab coefficients can be given in following table:

Table 3: Comparison of LUT partitions

Partition	Memory Locations	Partial Tables
No Partition	216=65536	0
8 Partition	2*(28)=512	2(each with 8 inputs)
4 Partition	4*(24)=64	4(each with 4 inputs)
2 Partition	8*(22)=32	8(each with 2 inputs)

5. PERFORMANCE ANALYSIS:

The results of FIR filter realization using LUT partitioning in terms of area with respect to various taps are shown below:

Table 4: Performance Analysis – Area

Filter tap	Partition	No. of Slice LUT	No. of Slice Registers
4	No	70	38
	8	-	-
	4	-	-
	2	85	7

8	No	426	218
	8	-	-
	4	131	20
	2	187	38
16	No	875	487
	8	151	7
	4	281	66
	2	429	77

From the above table we observe that the number of slice LUT and the number of slice registers both reduce with increase in number of LUT partitioning.

Table 5: Performance Analysis – Delay

Filter tap	No Partition	Partition 8	Partition 4	Partition 2
4	10.091	-	-	4.794
8	11.976	-	7.073	7.023
16	13.267	7.099	7.51	7.631

From the above table it can be observed that the delay is reduced when FIR filter with LUT partitioning is realized.

6. CONCLUSION AND FUTURE WORK

The Finite Impulse Response filter realization is done using Distributed Arithmetic then LUT partitioning is used to reduce the size of Lookup Tables. So the FIR filter provides reduction in access time compared to conventional method. The modified Distributed arithmetic reduces the area and delay significantly. The performance of FIR filter can be increased using pipelining.

REFRENCES

[1] T. Saramaki, “Finite Impulse Response Filter Design,” in Handbook for Digital signal Processing, edited by S. K. Mitra and J. F. Kaiser, J. Wiley & Sons, pp. 155-277, 1993.

[2] Keshab K. Parhi, “VLSI Digital Signal Processing Systems-Design and Implementation” edited by John Wiley and Sons, 1999.

[3] Ramesh R., Nathia R., “Realization of FIR Filter using Modified Distributed Arithmetic Architecture” Signal & Image Processing: An International Journal (SIPIJ) Vol.3, No.1, February 2012.

[4] Jiafeng Xie*, Jianjun He, Guanzheng Tan, “FPGA realization of FIR filters for high speed and medium speed by using modified distributed arithmetic architectures” Microelectronics Journal 41 (2010) 365-370.

[5] M. Yazhini, R. Ramesh, “FIR Filter Implementation using Modified Distributed Arithmetic Architecture”, Indian Journal of Science and Technology,2013.

[6] Lingjuan Wu*, Yingying Cui, Jie Huang, Dunshan Yu Institute of Microelectronics, Electronics Engineering and Computer Science School ,Peking University, Beijing.

[7] Burhan Khurshid, Roohie Naz Mir, “A hardware Intensive approach for Efficient Implementation of Numerical Integration for FPGA platforms”, at 27th International conference on VLSI Design and 13th International Conference on Embedded systems,2014.