IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 4, Issue 3, Ver. III (May-Jun. 2014), PP 01-09
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

Area Efficient Implementation of FIR Filter Using Distributed Arithmetic with Offset Binary Coding

D.Kalaiyarasi[1], T.Kalpalatha Reddy[2]
[1] *(Research Scholar, Department of Electrical and Electronics Engineering, Sathyabama University, Chennai, India)*
[2]*(Dean and Research, Department of Electronics and Communication Engineering, S.K.R Engineering College, Chennai, India)*

Abstract: *This paper presents the realization of area efficient architecture using Distributed Arithmetic with Offset Binary Coding (DA-OBC) for implementation of Finite Impulse Response (FIR) Filter. Area complexity in the algorithm of Finite Impulse Response Filter is mainly caused by multipliers. Among the multiplierless techniques of FIR Filter, Distributed Arithmetic is most preferred area efficient technique. In this technique, partial products of filter coefficients are precomputed and stored in Lookup Table (LUT) and the filtering is done by shift and accumulate operations on these partial products. However, the scale of the LUT will increase exponentially with the co-efficient. If the co-efficient is small, it is very convenient to realize. While the co-efficient is large, it will take up a lot of storage resources of FPGA and reduce the calculation speed. The paper presents the improvement of the DA algorithm by reducing the LUT size and delay using Offset Binary Coding Algorithm. The design based on Altera EP2S15F484C3 Chips is synthesized under the integrated environment of Quartus II 9.1. The result of simulation and test shows that Offset Binary Coding greatly reduces the FPGA hardware resources and the high speed filtering is achieved compared to conventional DA Algorithm.*

Keywords: *Finite Impulse Response (FIR); Distributed Arithmetic (DA); Lookup Table (LUT); Offset Binary coding (OBC); Field Programmable Gate Arrays (FPGA).*

I. Introduction

Due to the intensive use of FIR filter in video and communication systems, high performance in speed, area and power consumption is demanded. Basically, digital filters are used to modify the characteristics of signals in time and frequency domain and have been recognized as primary digital signal processing [1]. There has been a growing trend to implement DSP functions in FPGA last few years, which offer a balanced solution in comparison with Application Specific Integrated Circuits (ASICs) and Digital Signal Processors (DSPs). The advantages of the FPGA approach to digital filter implementation include high sampling rates than are available from tradition DSP chips, lower costs than ASIC for moderate volume applications. In that sense, the research community has put great effort in designing efficient architectures for DSP functions such as finite impulse response filters which are extensively used most of signal processing application such as Digital Communication, Speech Processing, Wireless/Satellite Communication, Bio-medical Signal Processing and many others due to its linearity and stability. Only the limitation offer by it is large number of taps, to get desired frequency response, which leads to area complexity. In general form, the FIR filter [2] is characterized by

$$Y[n] = h[n] * x[n] \qquad (1)$$

$$Y[n] = \sum_{k=0}^{N-1} h(k)x(n-k) \qquad (2)$$

where x[n] = input sequence
h[n] = impulse response of the filter

Equation (2) shows that, output of FIR filter is sum of product of impulse response and input sequence i.e. the extensive sequence of multiplication operations. Since the multipliers are costly in terms of area, several multiplierless schemes had been proposed. These methods can be classified in two categories based on how filter co-efficients are manipulated for the multiplication operation. The first category of multiplerless technique is the conversion based approach, in which filter co-efficients are transformed into other numerical representations whose hardware implementation or arithmetic manipulation is more efficient than the traditional binary representation. One of the examples of such a multiplierless technique is Canonic Signed Digit (CSD) method, in which filter co-efficients are represented by a combination of power of two in such a way that multiplication can be implemented simply by adder/subtractor and shifter [3]. Dempster-Mcleod method is also conversion based

approach but in this case partial results are arranged in cascade to get further savings in the usage of adders [4]. Second category of multiplierless technique is memory based approach involves memory or lookup tables used to store precomputed filter co-efficient operations. Constant co-efficient multiplier [5] and Distributed Arithmetic [6] are the memory based methods.

Distributed Arithmetic (DA) is the one of the efficient technique for realization of higher order filters as it can achieve high throughputs without the help of a hardware multiplier which was first developed by Croisier et.al [7]. The complicated multiplication-accumulation operation is converted to the shifting and adding operation when DA algorithm is directly applied to realize FIR filter [8]. The DA has proved to be an area efficient technique of FIR filter implementation. While using this technique special care is required against exponential growth of LUT size. Slicing of LUT to the desired length, gives an effective solution [9].

In this paper area efficient implementation of FIR filter using Distributed Algorithm with Offset Binary Coding is proposed. The proposed algorithm reduces the LUT size by a factor of 2 to 2^{N-1}. The obtained implementation results of proposed method are compared with conventional DA algorithm. The next section describes DA based FIR filter and section 3 describes DA with offset binary coding based FIR filter. Section 4 gives the performance analysis of FIR filter using proposed algorithm.

II. Distributed Arithmetic Based FIR Filter

Distributed arithmetic is a bit level rearrangement of a multiply accumulate to hide the multiplications. It is a powerful technique for reducing the size of a parallel hardware multiply-accumulate that is well suited to FPGA designs. At any instant n, the output y[n] of N-tap FIR filter is given as

$$Y[n]=\sum_{i=0}^{N-1} h(i)x(n-i)$$

In simplified form,

$$Y = \sum_{i=0}^{N-1} h_i x_i \qquad (3)$$

Where $\{h_i\}$'s are M-bit filter co-efficients and $\{x_i\}$'s are input samples coded as k-bit 2's complement numbers given by

$$x_i = -x_{i,k-1} + \sum_{j=1}^{k-1} x_{i,k-1-j} 2^{-j} \qquad (4)$$

By substituting equation (4) in equation (3), we get

$$Y = \sum_{i=0}^{N-1} h_i (-x_{i,k-1} + \sum_{j=1}^{k-1} x_{i,k-1-j} 2^{-j}) \qquad (5)$$

$$Y = -\sum_{i=0}^{N-1} h_i x_{i,k-1} + \sum_{j=1}^{k-1} (\sum_{i=0}^{N-1} h_i x_{i,k-1-j}) 2^{-j} \qquad (6)$$

Let

$$h_{k-1-j} = \sum_{i=0}^{N-1} h_i x_{i,k-1-j} \quad \text{for } j = 0 \qquad (7.1)$$

and

$$h_{k-1} = -\sum_{i=0}^{N-1} h_i x_{i,k-1} \qquad (7.2)$$

Then equation (6) becomes

$$Y = \sum_{j=0}^{k-1} h_{k-1-j} 2^{-j} \qquad (8)$$

Therefore, by interchanging the summing order of i and j, the initial multiplications in equation (3) are now distributed to another computation pattern. Since the term h_j depends on the $x_{i,j}$ values and has only 2^N possible values, it is possible to precompute them and store them in a read only memory(ROM) or LUT. An input set of N bits ($x_{0,j}$, $x_{1,j}$, $x_{2,j}$,…….$x_{N-1,j}$) is used as an address to retrieve the corresponding h_j values. These intermediate results are accumulated in k clock cycles to produce one y value. This leads to a multiplier – free

realization of vector multiplication. Table 1 shows the content of the ROM for N=5. Fig.1. shows the DA implementation of a N-tap FIR Filter.

Table:1 Content of the ROM (N=5)

$x_{0,j}$	$x_{1,j}$	$x_{2,j}$	$x_{3,j}$	$x_{4,j}$	Content of the ROM or LUT
0	0	0	0	0	0
0	0	0	0	1	h4
0	0	0	1	0	h3
0	0	0	1	1	h3+ h4
0	0	1	0	0	h2
0	0	1	0	1	h2+ h4
0	0	1	1	0	h2+ h3
0	0	1	1	1	h2+ h3+h4
0	1	0	0	0	h1
0	1	0	0	1	h1+ h4
0	1	0	1	0	h1+ h3
0	1	0	1	1	h1+ h3+h4
0	1	1	0	0	h1+ h2
0	1	1	0	1	h1+ h2+h4
0	1	1	1	0	h1+ h2+h3
0	1	1	1	1	h1+ h2+h3+h4
1	0	0	0	0	h0
1	0	0	0	1	h0+h4
1	0	0	1	0	h0+h3
1	0	0	1	1	h0+h3+ h4
1	0	1	0	0	h0+h2
1	0	1	0	1	h0+h2+ h4
1	0	1	1	0	h0+h2+ h3
1	0	1	1	1	h0+h2+ h3+h4
1	1	0	0	0	h0+h1
1	1	0	0	1	h0+h1+ h4
1	1	0	1	0	h0+h1+ h3
1	1	0	1	1	h0+h1+ h3+h4
1	1	1	0	0	h0+h1+ h2
1	1	1	0	1	h0+h1+ h2+h4
1	1	1	1	0	h0+h1+ h2+h3
1	1	1	1	1	h0+h1+ h2+h3+h4

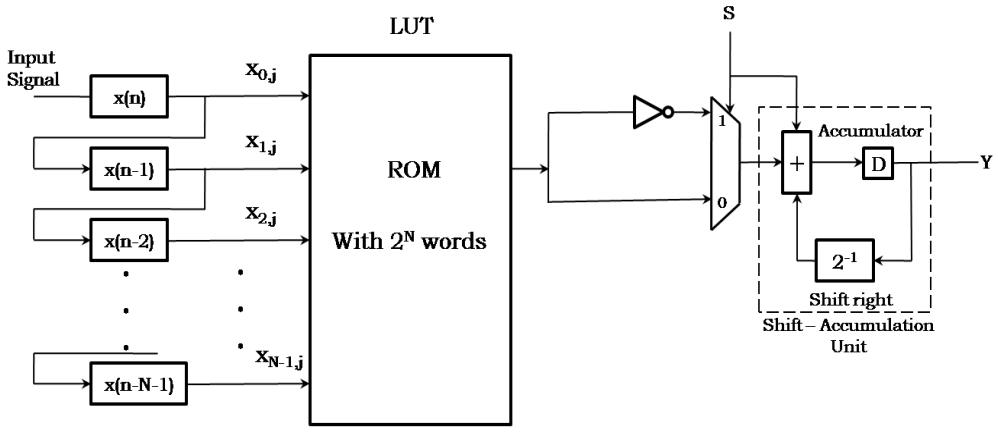


Fig 1. DA based implementation of N-tap FIR Filter

The Shift – Accumulate is a bit – parallel carry – propagate adder that adds the LUT content to the previous accumulated result. The inverter and the MUX are used for inverting the output of the ROM inorder to compute h_{k-1} and the control signal S is 1 when j = k-1 and 0 otherwise. The computation runs from j = 0 to j = k-1.

III. Distributed Arithmetic with Offset Binary Coding Based FIR Filter

The size of ROM is very important for high speed and area efficiency. The size of ROM increases exponentially with each added input address line. The proposed Offset – Binary Coding can reduce the ROM size by a factor of 2 to 2^{N-1}.

By rewriting equation (4) as,

$$x_i = \frac{1}{2}[x_i - (-x_i)]$$

$$x_i = \frac{1}{2}[-(x_{i,k-1} - \overline{x_{i,k-1}}) + \sum_{j=1}^{k-1}(x_{i,k-1-j} - \overline{x_{i,k-1-j}})2^{-j} - 2^{-(k-1)}] \quad (9)$$

where

$$-x_i = \overline{x_{i,k-1}} + \sum_{j=1}^{k-1} \overline{x_{i,k-1-j}}\,2^{-j} + 2^{-(k-1)}]$$

Let

$$d_{i,j} = \begin{cases} x_{i,j} - \overline{x_{i,j}} & for\ j \neq k-1 \\ -(x_{i,k-1} - \overline{x_{i,k-1}}) & for\ \ j = k-1 \end{cases} \quad (10)$$

and

$$d_{i,j} \in \{-1,+1\}$$

By substituting equation (10) in equation (9), we get

$$x_i = \frac{1}{2}[\sum_{j=0}^{k-1} d_{i,k-1-j}\,2^{-j} - 2^{-(k-1)}] \quad (11)$$

By using equation (11), equation (4) can be written as

$$Y = \sum_{i=0}^{N-1}\frac{1}{2}h_i[\sum_{j=0}^{k-1} d_{i,k-1-j}\,2^{-j} - 2^{-(k-1)}] \quad (12)$$

$$Y = \sum_{j=0}^{k-1}(\sum_{i=0}^{N-1}\frac{1}{2}h_i d_{i,k-1-j})2^{-j} - (\frac{1}{2}\sum_{i=0}^{N-1}h_i)2^{-(k-1)} \quad (13)$$

Define

$$D_j = \sum_{i=0}^{N-1}\frac{1}{2}h_i d_{i,j} \quad for \quad 0 \le j \le k-1 \quad (14)$$

and

$$D_0 = -\frac{1}{2}\sum_{i=0}^{N-1}h_i \quad (15)$$

By using equations (14) and (15), equation(13) can be written as

$$Y = \sum_{j=0}^{k-1} D_{k-1-j}\,2^{-j} + D_0\,2^{-(k-1)} \quad (16)$$

Table:2 Content of The ROM with OBC for N=5

$x_{0,j}$	$x_{1,j}$	$x_{2,j}$	$x_{3,j}$	$x_{4,j}$	Content of the ROM or LUT
0	0	0	0	0	-(h0+h1+ h2+h3+h4)/2
0	0	0	0	1	-(h0+h1+ h2+h3-h4)/2
0	0	0	1	0	-(h0+h1+ h2-h3+h4)/2
0	0	0	1	1	-(h0+h1+ h2-h3-h4)/2
0	0	1	0	0	-(h0+h1- h2+h3+h4)/2
0	0	1	0	1	-(h0+h1- h2+h3-h4)/2
0	0	1	1	0	-(h0+h1- h2-h3+h4)/2
0	0	1	1	1	-(h0+h1-h2-h3-h4)/2

0	1	0	0	0	-(h0-h1+ h2+h3+h4)/2
0	1	0	0	1	-(h0-h1+ h2+h3-h4)/2
0	1	0	1	0	-(h0-h1+ h2-h3+h4)/2
0	1	0	1	1	-(h0-h1+ h2-h3-h4)/2
0	1	1	0	0	-(h0-h1- h2+h3+h4)/2
0	1	1	0	1	-(h0-h1- h2+h3-h4)/2
0	1	1	1	0	-(h0-h1- h2-h3+h4)/2
0	1	1	1	1	-(h0-h1- h2-h3-h4)/2
1	0	0	0	0	(h0-h1- h2-h3-h4)/2
1	0	0	0	1	(h0-h1- h2-h3+h4)/2
1	0	0	1	0	(h0-h1- h2+h3-h4)/2
1	0	0	1	1	(h0-h1- h2+h3+h4)/2
1	0	1	0	0	(h0-h1+ h2-h3-h4)/2
1	0	1	0	1	(h0-h1+ h2-h3+h4)/2
1	0	1	1	0	(h0-h1+ h2+h3-h4)/2
1	0	1	1	1	(h0-h1+ h2+h3+h4)/2
1	1	0	0	0	(h0+h1- h2-h3-h4)/2
1	1	0	0	1	(h0+h1- h2-h3+h4)/2
1	1	0	1	0	(h0+h1- h2+h3-h4)/2
1	1	0	1	1	(h0+h1- h2+h3+h4)/2
1	1	1	0	0	(h0+h1+ h2-h3-h4)/2
1	1	1	0	1	(h0+h1+ h2-h3+h4)/2
1	1	1	1	0	(h0+h1+ h2+h3-h4)/2
1	1	1	1	1	(h0+h1+ h2+h3+h4)/2

Table II Shows the Content of the ROM with OBC for N=5.In this table, D_j values are mirrored along the line between the 16th and 17th rows. In otherwords, the term D_j has only 2^{N-1} possible values depending on the $x_{i,j}$ values. Therefore it is possible to reduce the ROM size by a factor of 2. Table III shows the content of reduced ROM with OBC.

Table: 3 Content of reduced ROM with OBC (N=5)

$x_{1,j}$	$x_{2,j}$	$x_{3,j}$	$x_{4,j}$	Content of the ROM or LUT
0	0	0	0	$-(h_0+h_1+ h_2+h_3+h_4)/2$
0	0	0	1	$-(h_0+h_1+ h_2+h_3-h_4)/2$
0	0	1	0	$-(h_0+h_1+ h_2-h_3+h_4)/2$
0	0	1	1	$-(h_0+h_1+ h_2-h_3-h_4)/2$
0	1	0	0	$-(h_0+h_1- h_2+h_3+h_4)/2$
0	1	0	1	$-(h_0+h_1- h_2+h_3-h_4)/2$
0	1	1	0	$-(h_0+h_1- h_2-h_3+h_4)/2$
0	1	1	1	$-(h_0+h_1-h_2-h_3-h_4)/2$
1	0	0	0	$-(h_0-h_1+ h_2+h_3+h_4)/2$
1	0	0	1	$-(h_0-h_1+ h_2+h_3-h_4)/2$
1	0	1	0	$-(h_0-h_1+ h_2-h_3+h_4)/2$
1	0	1	1	$-(h_0-h_1+ h_2-h_3-h_4)/2$
1	1	0	0	$-(h_0-h_1- h_2+h_3+h_4)/2$
1	1	0	1	$-(h_0-h_1- h_2+h_3-h_4)/2$
1	1	1	0	$-(h_0-h_1- h_2-h_3+h_4)/2$
1	1	1	1	$-(h_0-h_1- h_2-h_3-h_4)/2$

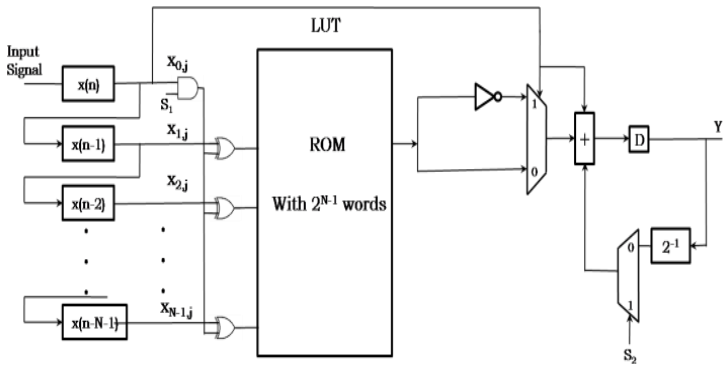


Fig. 2 DA with OBC based implementation of N-tap FIR Filter

Fig.2. shows the implementation of DA with OBC based N-tap FIR Filter. Again computation starts from LSB of x_i, i.e., j=0. The XOR gates are used for address decoding, the MUX with constant D_0 provides the initial value to the shift – accumulator and the MUX after the ROM is used to inverse the output of ROM when

j=k-1. Two control signal s_1 and s_2 are required, where s_1 is 1 when j = k-1 and 0 otherwise, and s_2 is 1 when j = 0 and 0 otherwise.

IV. Implementation Results

The design and implementation of the existing method and the proposed method is done using the Verilog HDL coding and synthesized on Altera, Quartus II 9.1. The Altera Quartus II design software provides a complete, multiplatform design environment that easily adapts to specific design needs. It is a comprehensive environment for system-on-a-programmable-chip (SOPC) design. The Quartus II software includes solutions for all phases of FPGA and CPLD design as shown in figure 3. In addition, the Quartus II software allows us to use the Quartus II graphical user interface and command-line interface for each phase of the design flow.

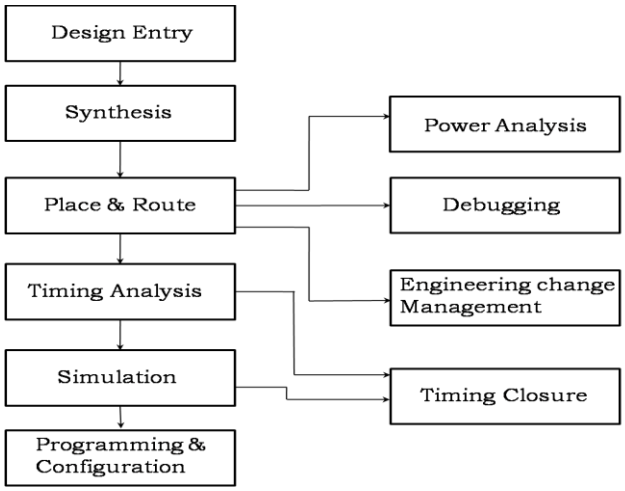
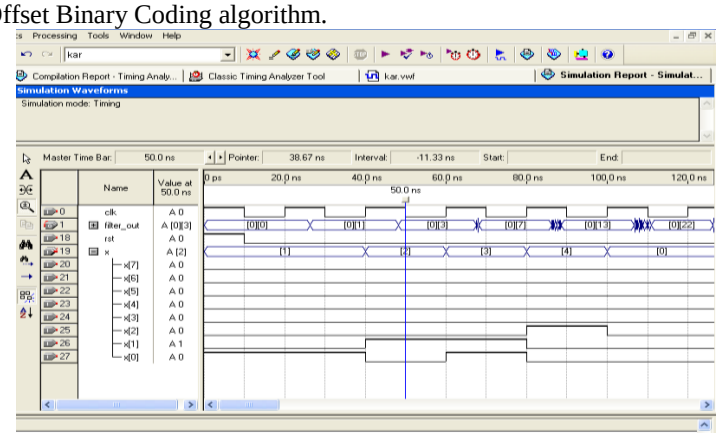


Fig. 3 Quartus II Design Flow

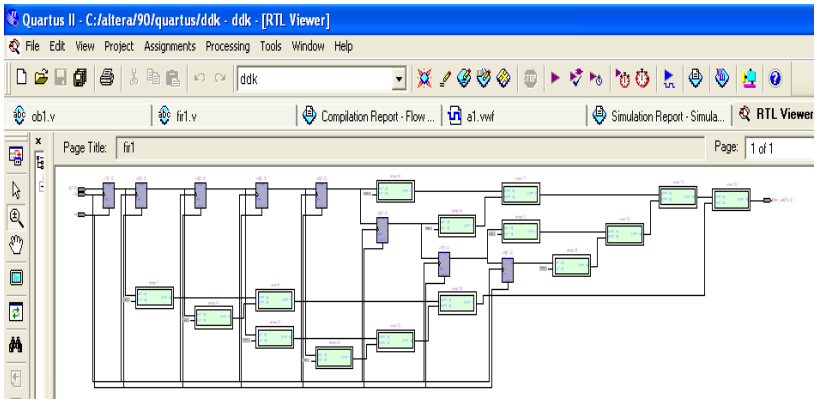
Design procedure:

Step 1: Derive the filter Co-efficient according to specification of filter.
Step 2: Store the input value in input register.
Step 3: Design the LUT, which represents all the possible sum combination of filter co-efficient.
Step 4: Accumulate and shift the value according to partial term beginning with LSB of the input and shift it to the right to add it to the next partial result.
Step 5: Analyze the output of filter as per specification.

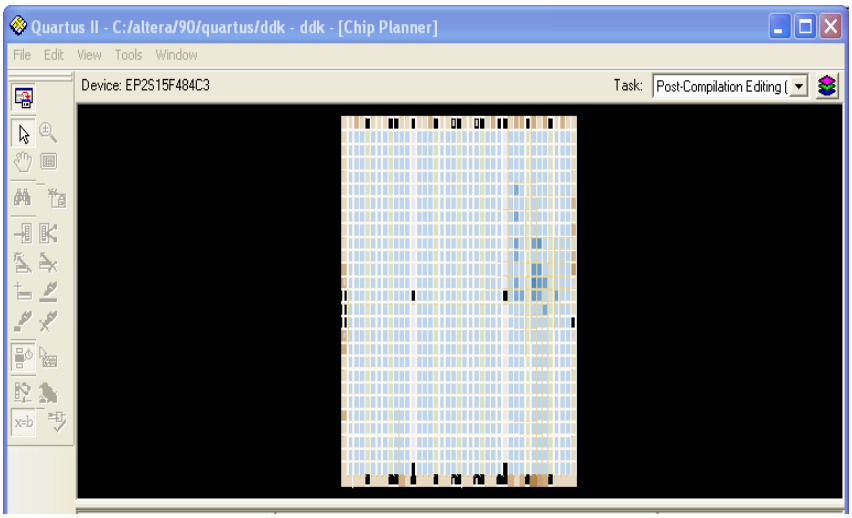
The 8-tap FIR filter is implemented for performance analysis of conventional method, DA algorithm and proposed DA with Offset Binary coding algorithm. The test bench simulation results for 8-tap conventional FIR filter is shown in fig 4(a). The RTL view and Chip layout of conventional FIR filter are shown in Fig 4(b) and 4(c). Fig 5(a), 5(b) and 5(c) show the test bench simulation, RTL view and Chip layout of 8-tap FIR filter using DA algorithm. Fig 6(a), 6(b) and 6(c) show the test bench simulation, RTL view and Chip layout of 8-tap FIR filter using DA with Offset Binary Coding algorithm.



(a) Test bench Result

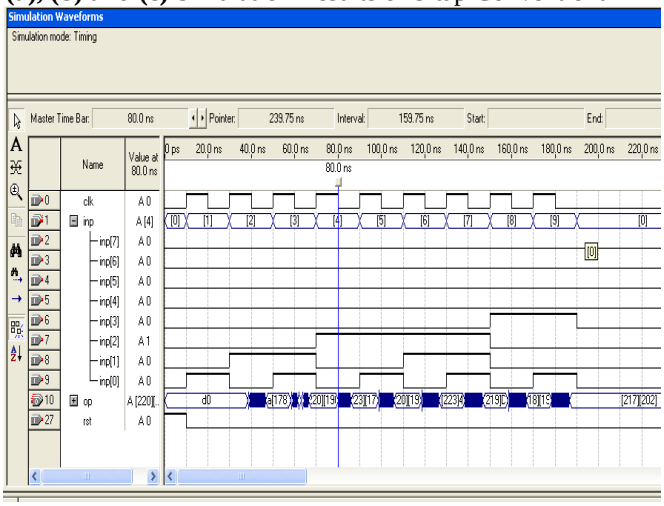


(b) RTL View

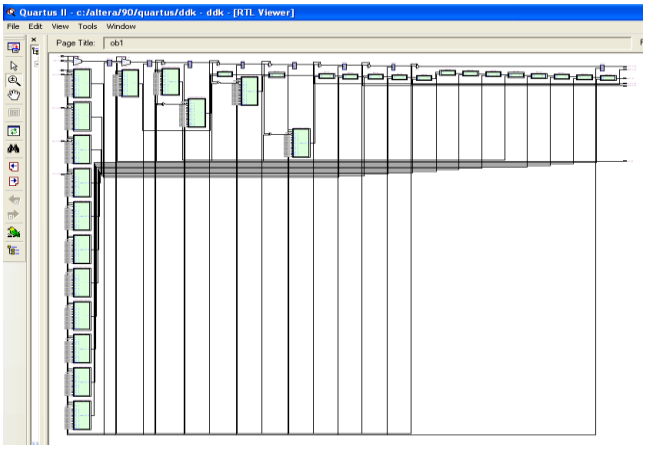


(c) Chip Layout

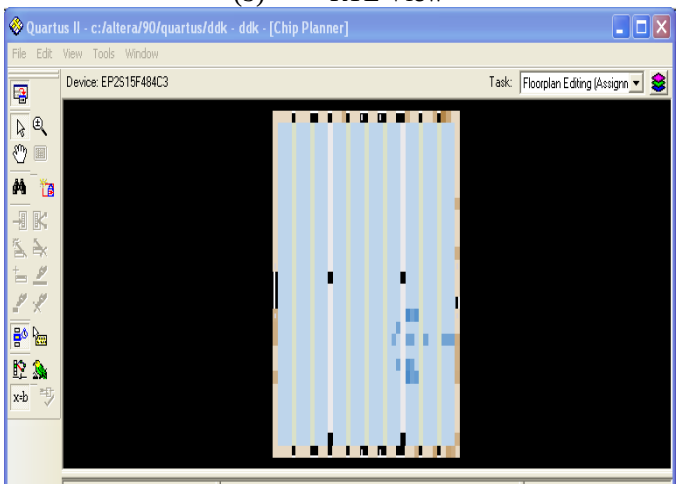
Fig. 4 (a), (b) and (c) Simulation Results of 8-tap Conventional FIR filter



(a) Test bench Result

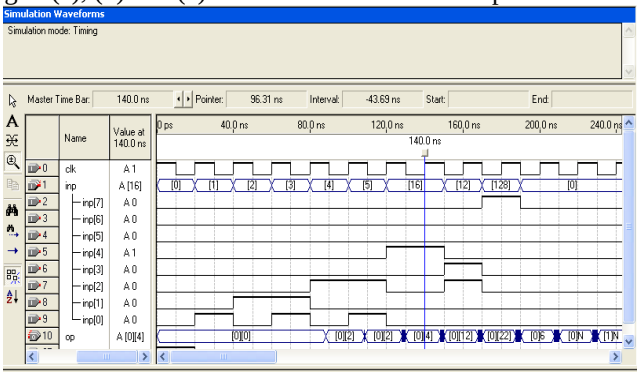


(b) RTL View

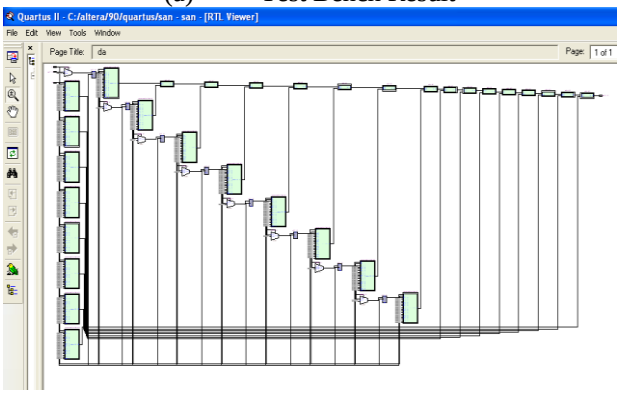


(c) Chip Layout

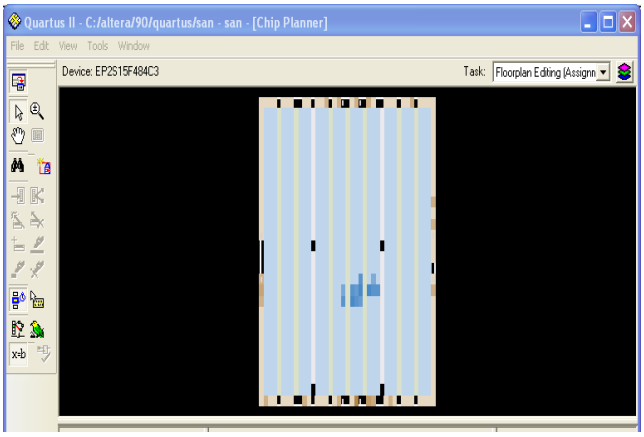
Fig. 5 (a), (b) and (c) Simulation Results of 8-tap DA FIR filter



(a) Test Bench Result



(b) RTL View



(c) Chip Layout

Fig.6 (a), (b) and (c) Simulation Results of 8-tap DA-OBC FIR filter

The implementation results of 8-tap FIR filter after applying the DA algorithm and DA-OBC algorithm as shown in Table IV.

Table: 4 Implementation Result

Parameter	Conventional FIR filter	DA FIR filter	DA-OBC FIR filter
Dedicated logic register	67	168	68
Delay (ns)	15.923	3.860	3.547
Power (mW)	324.4	324.41	324.4

Table IV shows that 8-tap DA-OBC FIR filter reduce the memory requirement and time delay. Therefore the proposed design utilizes very less chip area compared to DA algorithm, which stems from the fact that it demands half the memory size of DA algorithm and utilize less combinational logic. The proposed algorithm reduces the time delay very much as compared to conventional method.

V. Conclusion

The Complicated Multiplication – Accumulation operation is converted to the shifting and addition operation when the DA algorithm is directly applied to realize FIR filter. However, the size of LUT increases exponentially with each added input address line. The proposed algorithm for FIR filter synthesized under the integrated environment of Altera, Quartus II 9.1 which is area efficient since it reduced the memory requirement by a factor of 2 as compared to conventional DA algorithm and the proposed algorithm reduces delay approximately 5 times as compared to conventional FIR filter. So these filters can be used in various applications such as Adaptive filtering for noise cancellation and echo cancellation, pulse shaping in WCDMA, software design radio and signal processing system for high speed. In future the work to reduce the power consumption by reducing the critical path using either pipelining or parallel processing could be performed.

References

[1]. Mariusz Rawski, Pawel Tomaszewicz, Henry Selvaraj and Tadeusz Lub, "Efficient Implementation of Digital Filters with use of Adavanced Synthesis Methods Targeted FPGA Architecture", 8th Euromicro Conference on digital system design (DSD'05), IEEE Computer Society, 2005.

[2]. S.K. Mitra, "Digital Signal Processsing – a computer Based Approach", TATA McGraw Hill, Second edition, pp. 427-432,2008.

[3]. M.Yamada and A.Nishihara, "High – Speed FIR Digital Filter with CSD coefficients Implemented on FPGA" , in Proceedings of IEEE Design Automation Conference, 2001, pp.7-8.

[4]. Martinez – Peiro, J.Valls, T. Sansaloni, A.P. Pasual, and E.I.Boemo, " A Comparison between Lattice, cascade and Direct Form FIR filter structures by using a FPGA Bit – Serial DA Implementation", in proceeding of IEEE International Conference on Electronics, Circuits and Systems, 1999, Vol.1.pp.241-244.

[5]. S.S.Jeng, H.C.Lin, and S.M.Chang, " FPGA implementation of FIR filter using M-bit parallel distributed Arithmetic", IEEE ISCAS'2006, pp.875-878.

[6]. H.Yoo and D.V.Anderson, " Hardware – Efficient Distributed Arithmetic Architecture for Higher – order Digital filter", IEEE International Conference on Acoustic speech and Signal Processing, ICASSP, pp.12-128,2005.

[7]. A.Croisier, D.J. Esteban, M.E. Levilion and V.Rizo, " Digital filter for PCM encoded signals",U.S. Patent No.3, 777 130, issued april, 1973.

[8]. Magatha Nayak Bhukya, K. Anjaiah, G.Sravya, P. Wagaraju, " The Design of High speed FIR Filter using Improved DA Algorithm and its FPGA Implementation", International Journal of Engineering Trends and Technology, Vol. 3, Issue 2, 2012.

[9]. S.M.Badave and A.S.Bhalchandra, " Multiplierless FIR Filter Implementation on FPGA", Internation Journal of Information and Electronics Engineering, Vol. 2, No.3, May 2012.