Asic Implementation of Energy Efficient Reduced Size Array Multiplier

[1]K.Jayasurya,[2]M.Sangeetha
[1]*PG Scholar, Dr.Mahalingam college of Engineering and Technology, Pollachi,*
[2]*Assistant Professor, Department of Electronics and Electrical Engineering, Dr.Mahalingam college of Engineering and Technology, Pollachi,*

Abstract

A highly efficient carry free multiplication plays a vital role in arithmetic operations. A special addition mechanism employed in the proposed work processes the bits from higher order to lower order i.e. from most significant bit to least significant bit is used for improving the performance of the multiplier by reducing the critical path delay. This left to right mode operation results in the reduction of partial products. This is achieved by effectively using on-the fly conversion (OTFC) along with radix-4 full adders. It also results in discarding final addition in the most significant part during the ripple carry addition in the least significant part. This minimizes overall delay of the multiplier compared with the conventional partial product generation. The proposed LRRS (left-to-right reduced size) multiplier is implemented with optimized on-the fly conversion circuit to reduce the multiplier complexity. Compared with conventional left-to-right multipliers the proposed full length multiplier and its truncated version results in reduction of area, power and delay. Finite Impulse Response filter involves multiplications, additions and shifting operations. As the multiplier is the slowest element in the system, it will affect the performance of the FIR filter. In this case, the reduced size, energy-efficient left-to-right array multiplier is designed which reduces the area, power and delay of the multiplier unit. Therefore the area is reduced with the increase in speed of operation and low power consumption in the multiplier unit of FIR filter which serves as the Application Specific IC.

Keywords: *Left-to-right reduced size, on-the fly conversion, array reduction, truncated multiplier, FIR Filter.*

I. INTRODUCTION

Multiplication is an essential process applied in various signal processing algorithms. Multipliers normally occupy more area, with longer delay and more power consumption. Therefore low-power and flexible multiplier design plays a vital part in low-power VLSI signal processing design [1]-[3]. There has been number of work carried out on low-power multipliers at various levels in VLSI design flow. The overall performance of the entire signal processing unit depends mainly on the multipliers performance. Because the multiplier it comprises to 70% of the overall design. Furthermore, it is affected with the critical path delay in its partial product addition part. Hence, enhancement of various parameters of the multiplier is a major design task. One of the parameters, either area or speed needs to be compromised to better the other. There is always a conflicting interest between these parameters. In low-power multiplication design, many existing work are done based on minimizing the switching activities have been published [4]-[5]. Besides that, an easy way to design low power multiplier architecture is by creating power efficient full adder design in the partial product adder tree. The other way to minimize the switching activity is by effectively performing the operation of the operand when required or by using shift and add designs. Furthermore, the reduction of the power in multipliers is also possible through various row and column bypassing designs where the number of zeros is more in the operands.

Multiplication operation is normally performed via three stages. They are: generation of partial products or (PPG) by using AND gate or by encoding algorithms, addition of partial products (PPA) by full adders are compressor tree structures, and final stage addition with look-ahead or parallel prefix adders. There are also sequential multiplication implementations in various designs such as MAC (Multiply-Accumulate). We mainly focus on combinational part of the design especially partial product reduction part. Because the volume of grouping the design in integrated circuit grows in digital VLSI systems. Variety of multiplication algorithms differ in the approaches of partial product generation, reduction, and final stage addition. For PPG, radix-2 is the simplest form. To minimize the number of partial product the operands are basically coded in the radix representation. The other important type is the radix-4 digit set comprised of {-2,-1, 0, 1, 2}[6]. For partial product reduction, two ways exist: reduction by partial product rows, executed by a group of adders, and reduction by columns in the partial product, done by counting operation. The final stage adder requires a fast addition scheme because it is on the carry propagate path. In certain cases, final addition is postponed and those partial results are used for redundant operations.

Our objective is to improve the performance and optimize the power of carry free left-to-right multipliers proposed in [6] and [7]. Basic array multiplier has simple design compared with compression adder multipliers. Left-to-right has certain unique properties.
1. The partial product in MSB part follows the addition similar to redundant adders.
2. On-the-fly conversion is used in the MSB side which produces the result during the same time duration as that of LSB.
3. Glitch minimization is achieved by reducing the propogation.

This is further enhanced with breaking the linear structure present in the conventional LRCF multiplier. This proposed multiplier called as LRRS multiplier is designed with radix-4 architecture. This can be further extended to truncated programmable multiplier. The proposed multiplier results in low PDP with less complexity.

In Section II the basic LRCF multiplier along with redundant on-the-fly conversion part (OTFC) is described. The proposed left-to-right reduced-size multiplier (LRRS) along with its radix-2 and radix-4 type is reviewed in Section III and its full length and truncation implementation is discussed in Section IV. Evaluation results are analyzed in Section V. In section VI FIR filter implementation is discussed. Paper is concluded In Section VII along with its future possible enhancement.

II. BASIC LEFT-TO-RIGHT ARRAY MULTIPLIER

Basic multiplication with n-bit operands in the range [0, 1) describing product p = x × y is described in equation(1).

$$x = \sum_{i=1}^{n} x_i 2^{-i}, \quad y = \sum_{i=1}^{n} y_i 2^{-i}, x_i,\ y_i \in \{0,1\} \qquad (1)$$

In a general multiplier k-bit adder is used in the MSB part to get the first half of the multiplier output. By controlling the column rounding and truncation is implemented in the multiplier. Fig.1 shows the basic architecture of Left to Right Multiplier along with OTFC conversion circuit [6]. It avoids the final separate addition present in the MSB part.

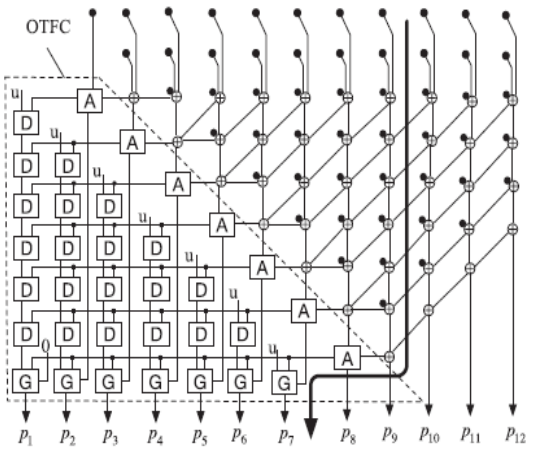


Fig.1 Left to Right Multiplier

The operation of on-the-fly conversion is performed parallel and array reduction also achieved, thus reducing delay compared with conventional designs. Carry digit representation has a sum and carry pair (*c*, *s*) during the MSB part reduction process. The condition based analysis is done depending on the carry generation. For an efficient carry free process the redundant representation logic is followed. This reduces the overall size of OTFC converter circuit. AS shown in figure, the black circle represents the AND gate used to generate the partial product. The bold line between P9 and P10 represents the critical path. The MSB part covered by the dotted line represents OTFC circuit. It consists of A,D and G cells. A cell takes the sum and carry from previous stage as the input and produces the sum(Zi,1) and carry(Zi,0) as its output. D cell is used to process the carry bit from the lower to higher bit part. It consists of AND gate to produce Dpi and NAND, NOR, INV to produce Dgi. The G cell is used in the final stage and it is similar to D cell with NAND, NOR, INV gates along with xor gate to produce the final product. The internal block of A,D and G is shown in Fig.2.

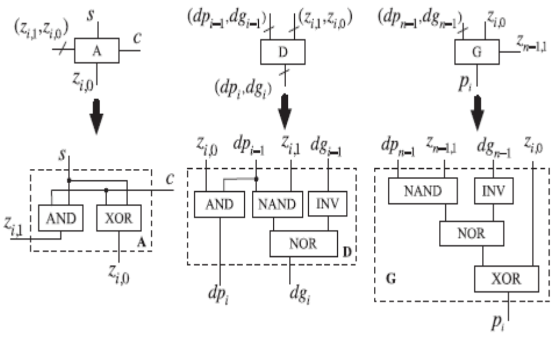


Fig.2 Internal Design of A, D and G cells

III. PROPOSED LEFT-TO-RIGHT REDUCED-SIZE (LRRS) ARRAY MULTIPLIER

In the conventional LRCF multiplier the partial product are produced from the MSB part. For increasing its performance, on-the-fly converter (OTFC) is used to minimize the carry propagation. This leads to increase in the overall area due to the OTFC circuit consisting of A, D, and G cells. The OTFC area plays a vital role in LRCF design. To reduce the overall multiplier size OTFC should be optimized [8].

Fig. 3 describes the block diagram of *n*-bit LRRS multiplier. The black dots indicates the AND gates producing the partial products. The module in the dotted line describes the left-to –right adder which is similar to that proposed in [7] except that our design does not need an on-the-fly adder (OTFA) in the least significant block.

This modified OTFC design produces the reduced-size multiplier. Compared to the A, D, and G cells in the conventional design proposed uses half-adder, Conditional adder and selection block. The overall reduced size multiplier is divided into two blocks BI and BII. BII consists of conventional LSB partial product generation and addition. Block BI consists of OTFC block with conditional-sum as a main unit. It is further divided into B1, B2 and B3 consisting of Half adder, Conditional adder and selection block as its corresponding units. Each sub-unit generates to conditional sum with one expecting carry and other expecting no carry. The final MSB output is obtained from the selection block which gets the input from the conditional adder. The conditional adder present in MSB part consisting of a ripple-carry adder and an incrementer.

A. *Radix-2 LRRS*

In the radix-2 LRRS multiplication the final MSB output produces n bits in which (n-1) are produced from BI block as shown in Fig.3. BI block with its sub-block B1, B2, B3 produces P1 to P7 and Bock BII generates P8 to P12 bits respectively. With radix-2 implementation LRRS is same as the conventional design with changes in the OTFC circuit. In fig.3 the design is based on left-to-right reduced size array based on the radix-2 method. The block BI is subdivided into B1, B2, B3 sub blocks to increases the precision based on the Wallace structure arrangement most significant part which is explained in section IV. Block BI receives the carry save outputs from the reduction array and forms two conditional sums. Block BII produces the carry into block BI.

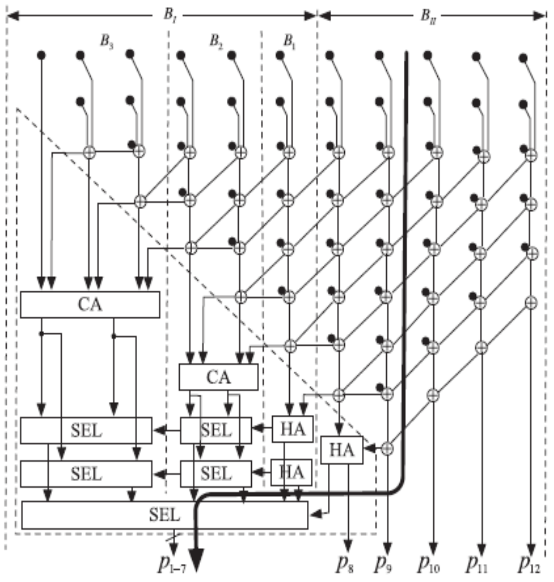


Fig.3 Left-to-right reduced size Multiplier

B. *Radix-4 LRRS*

Radix-4 implementation of LRRS multiplier is used to minimize the partial product generation, which optimizes the overall complexity and delay of the design. Separate radix-4 recoding structure is used to produce the reduced partial product. It uses the reduced digit set {−2,−1, 0, 1, 2}. The recoding structure is shown in Fig.4 generating PPij based on sign, one and two bit signals[2].

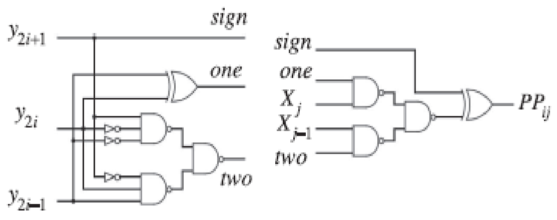


Fig.4 Radix-4 recoding

The complete architecture of radix-4 LRRS is shown in Fig.5. It shows the reduction in AND gate and adders present in the conventional LRRS design. It produces the conditional radix-4 adder output based on RCA and incrementer. The multiplier value is cascaded with 0 if n is odd or 1 if n is even. The sign extension can also be optimized using this design. The redundant bits are produced in each stage of the addition operation. The additions in the sub-blocks are performed in parallel with the array reduction, addition, and selections.

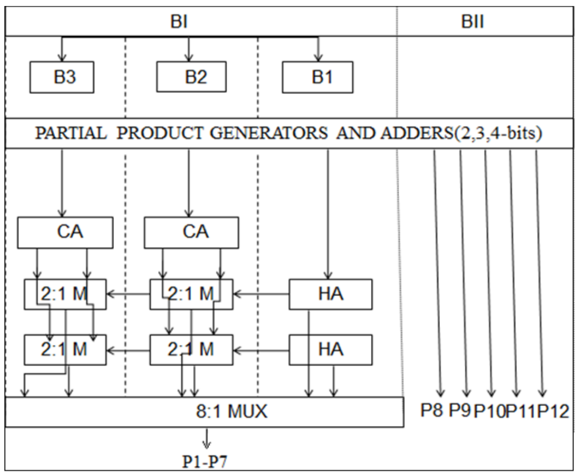


Fig.5 Implementation of radix-4 LRRS multiplier

The block BI is subdivided into B1, B2, B3 sub blocks to increases the precision. The additions in the sub-blocks are performed in parallel with the array reduction, addition, and selections. In LRRS multiplier, the partial products are generated and added. The sum is given to another adder receiving new partial products and carry is given to next array. The overlapped partial products are reduced and the final addition gives output. The normal full adders are replaced with approximate adders so that num of gates can be reduced.

IV. FULL PRECISION AND TRUNCATED MULTIPLIERS

The full precision multiplier based on radix-2 and radix-4 is implemented in previous section. Its overall area and power is analyzed in next section. The full length multiplier based on radix-4 results in optimized area. The LRRS truncation multiplier can be used easily to develop full length multiplier design. The overall delay is same for full length and reduced precision with gates getting reduced in the reduction part. The recoder and converter is also same for both the designs. For the LRRS_r4 design, the radix-4 full adder is needed to produce the LSB during the process of array reduction to minimize the extra delay due to the expansion to full precision.The array reduction process needs 0.5n2 − 2n+ 2 FA, n/2 modified HA, 1.5n − 2 regular HA, and n/2 + 1 FA_r4. Then the equivalent size of radix-4 LRRS full precision multiplier is 6.75n2 + 20.7n+ 29.5.

A. Error Analysis

Error analysis and error minimization methods of truncated design multipliers have been frequently analyzed in various existing papers. Certain variable error minimization methods
Significantly improve the accuracy of truncated multipliers by reducing errors. There are certain programmable truncated multipliers used to minimize area based on truncation column selection bit. Fig.6 shows the segmented analysis of truncated multiplier.

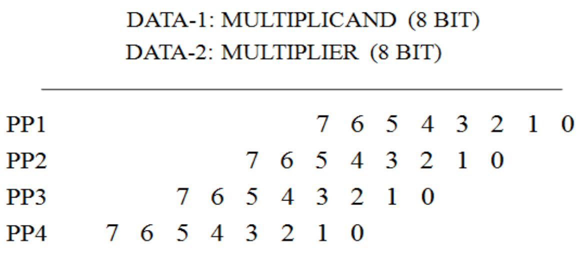


Fig.6.a Partial Product Generation based on radix-4 method

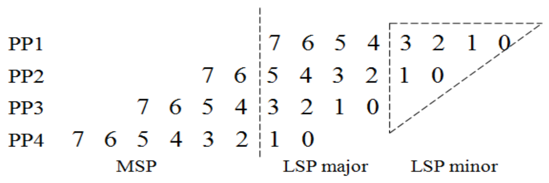


Fig.6.b Parting MSP and LSP

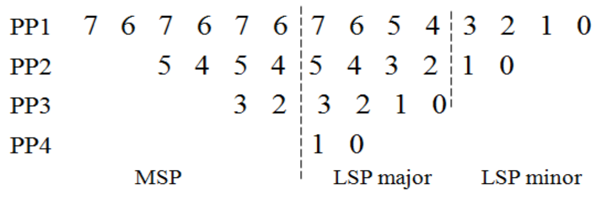


Fig.6.c Final Arrangement

The k columns present in the LSB part is considered as LSP major. One in the right most part is considered as LSP minor. The one which cascades LSP major and minor is called IC (input correction block). The MSP part is subdivided into three blocks (fig.5) based on the truncation arrangement. It is used to manage the error rate by compensation bits. Similarly the bits can be extended as 16, 32, 64, etc. The LRRS design with and without truncation is designed and its results are analyzed in next section for 8x8 and 16x16 radix-4 LRRS array multiplier.

V. EVALUATION RESULTS

It is inferred that the Table 1 shows that various parameters like LUT, Slices, power and delay have been analyzed and the comparative results of full length multiplier furnish the best performance compared with existing LRRS multiplier.

Table.1. Resource Utilization for LRRS full length Multiplier

Design Summary	Existing 8X8	Proposed 8X8	Existing 16X16	Proposed 16X16
LUT	52	47	329	315
Slices	34	27	174	161

Power(mW)	252	250	305	302
Delay(ns)	3.927	3.793	3.996	3.793

The Fig.7 shows the RTL simulation of radix-4 LRRS array multiplier and Fig.8 shows the RTL schematic of radix-4 LRRS array multiplier when the design is implemented in Xilinx ISE software.

The Fig.9 shows the output result of the design summary window obtained in XilinxISE. It contains the total number of various hardware resource utilized for the new proposed technique for full length multiplier shown in the Table 1. The design is synthesized using Xilinx FPGA.

From the Fig.10 shows the power report of the new proposed design obtained in Xilinx ISE. It contains the total power consumed for the proposed full length multiplier as mentioned in the Table 1. A final observation is that compared with LRCF design the proposed LRRS multiplier results in better area and power. It is further improved in truncated design.

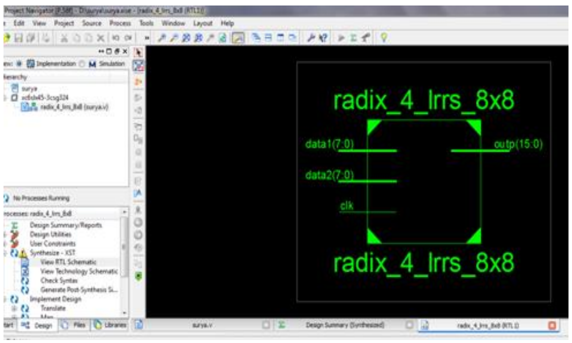


Fig.7. RTL Simulation of Radix-4 8x8 LRRS Array Multiplier

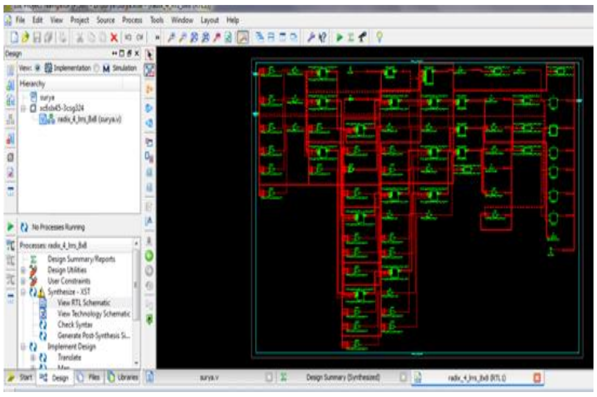
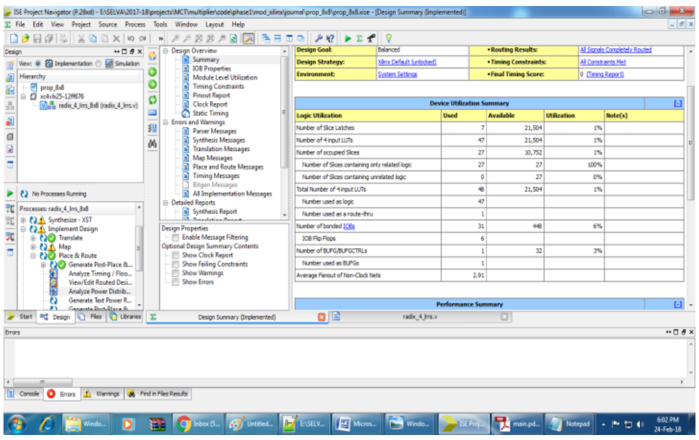


Fig.8. RTL Schematic of Radix-4 LRRS Array Multiplier



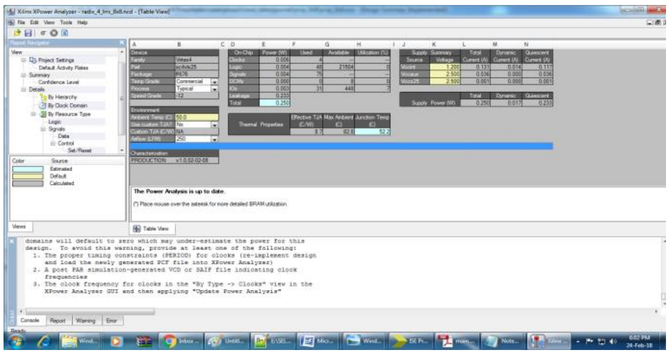
Design Goal:	Balanced	•Routing Results:	All Signals Completely Routed
Design Strategy:	Xilinx Default (unlocked)	•Timing Constraints:	All Constraints Met
Environment:	System Settings	•Final Timing Score:	0 (Timing Report)

Device Utilization Summary

Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Latches	7	21,504	1%	
Number of 4 input LUTs	47	21,504	1%	
Number of occupied Slices	27	10,752	1%	
Number of Slices containing only related logic	27	27	100%	
Number of Slices containing unrelated logic	0	27	0%	
Total Number of 4 input LUTs	48	21,504	1%	
Number used as logic	47			
Number used as a route-thru	1			
Number of bonded IOBs	31	448	6%	
IOB Flip Flops	6			
Number of BUFG/BUFGCTRLs	1	32	3%	
Number used as BUFGs	1			
Average Fanout of Non-Clock Nets	2.91			

Performance Summary

Fig.9. Design summary of radix-4 LRRS Multiplier



The Power Analysis is up to date.

(*) Place mouse over the asterisk for more detailed BRAM utilization.

```
domains will default to zero which may under-estimate the power for this
design.  To avoid this warning, provide at least one of the following:
  1. The proper timing constraints (PERIOD) for clocks (re-implement design
     and load the newly generated PCF file into XPower Analyzer)
  2. A post PAR simulation-generated VCD or SAIF file indicating clock
     frequencies
  3. The clock frequency for clocks in the "By Type -> Clocks" view in the
     XPower Analyzer GUI and then applying "Update Power Analysis"
```

Fig.10. Power analysis of radix-4 LRRS Multiplier

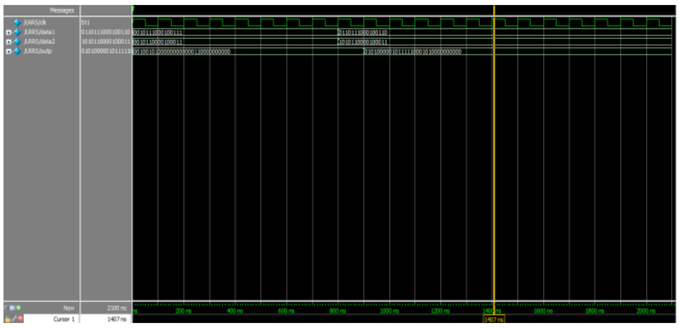


Fig.11. Simulation Results for radix-4 LRRS Multiplier

The Fig. 11 and shows the output result of the full length output with data1 and data2 as the multiplier input with output signal as its full length output.

VI. IMPLEMENTATION IN FIR FILTER (4-TAP)

Finite Impulse Response filters are the most important element in signal processing and communication and its features are explained in section 1.2. Area optimization and speed are the key requirements of Finite Impulse Response filters. Finite Impulse Response filter involves multiplications, additions and shifting operations. As the multiplier is the slowest element in the system, it will affect the performance of the FIR filter. Finite Impulse Response filter based on LRRS array multiplier is designed and compared with conventional filter to analyze the best result.

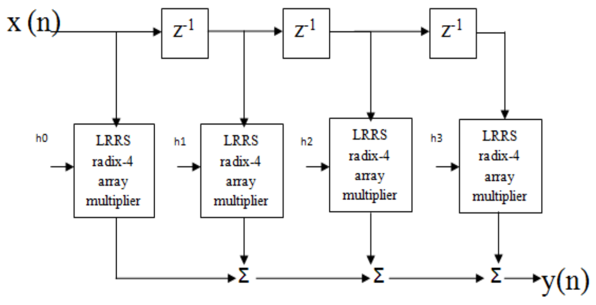

x (n)
z-1
z-1
z-1
h0
h1
h2
h3
LRRS radix-4 array multiplier
LRRS radix-4 array multiplier
LRRS radix-4 array multiplier
LRRS radix-4 array multiplier
Σ
Σ
Σ
y(n)

Fig.12 4-Tap FIR Filter Using LRRS Radix-4 Array Multiplier

The designed multiplier is implemented in cadence Encounter and comparison is obtained.

Table 2 Comparison of different LRRS Multiplier

MULTIPLIER TYPE	AREA	POWER(n W)	DELAY(p s)
Modified LRRS R-4 array	1930	672521.353	1833
Existing LRRS R-4 array	2053	813215.56	1935

The RTL schematic of designed multiplier implemented in 4-tap FIR filter is shown in fig.13

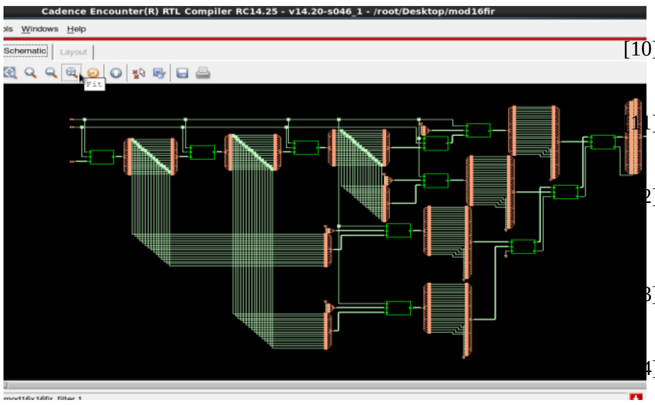

Cadence Encounter(R) RTL Compiler RC14.25 - v14.20-s046_1 - /root/Desktop/mod16fir
Windows
Help
Schematic
Layout
Fit

Fig 13 RTL Schematic of FIR filter

VII.CONCLUSION AND FUTURE WORK

A left-to-right reduced-size carry-save array multiplier (LRRS) is implemented for both full length and truncated version. The MSB part is processed before the LSB part leading to lesser delay. Further, optimized on-the-fly-conversion (OTFC) with radi-4 design and conditional adder results in reduction in area and power. Compared with existing the proposed method results in 10% area reduction with 5% power reduction. In the future, the use of this multiplication approach in squaring, sum of squares operations and other signal processing applications can be analyzed.

REFERENCES

[1] Z. Shun, O. A. Pfander, H. J. Pfleiderer, and A. Bermak, "A VLSI architecture for a run-time multi-precision reconfigurable booth multiplier," in Proc. 14th IEEE Int. Conf. Circuits Syst., pp. 975–978, 2007.

[2] K.Sucharitha, P. Rahul Reddy,"An Efficient Implementation of Multipliers for ASIC Implementations",International Journal of VLSI & Signal Processing (SSRG-IJVSP),Volume3 Issue 1, 2016.

[3] M. de la Guia Solaz, W. Han, and R. Conway, "A flexible low power DSP with a programmable truncated multiplier," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 59, no. 11, pp. 2555–2568, 2012.

[4] Y.RamaLakshmanna, G.V.S.Padma Rao, N . Udaya Kumar, K. Bala Sindhuri," A Survey on Different Multiplier Techniques" International Journal of Electronics and Communication Engineering (SSRG - IJECE),Volume 3 Issue 3 2016.

[5] J. Chen, C. H. Chang, F. Feng, W. Ding, and J. Ding, "Novel design algorithm for low complexity programmable FIR filters based on extended double base number system," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 62, no. 1, pp. 224–233, 2015.

[6] Surendra Verma, Dr. K C Mahajan,"A Survey on Energy Efficient Routing Protocols for Wireless Sensor Networks & Comparative analysis with USEP",International Journal of Electronics and Communication Engineering, Volume-3 Issue-10 2016.

[7] M. C. Wen, S. J. Wang, and Y. N. Lin, "Low power parallel multiplier with column bypassing," in Proc. IEEE Int. Symp. Circuits Syst., pp. 1638–1641, 2005.

[8] M. Ahuja and Sakshi, "Design and analysis of bypassing multiplier," in Proc. 5th Int. Conf. ARTCom, pp. 241–246, 2013.

[9] M. D. Ercegovac and T. Lang, "Fast multiplication without carrypropagate addition," IEEE Trans. Comput., vol. 39, no. 11, pp. 1385–1390, Nov. 1990.

[10] L. Ciminiera and P. Montuschi, "Carry-save multiplication schemes without final addition," IEEE Trans. Comput., vol. 45, no. 9, pp. 1050–1055, 1996.

1] M. D. Ercegovac and T. Lang, "On-the-fly conversion of redundant into conventional representation," IEEE Trans. Comput., vol. C-36, no. 7, pp. 895–897, 1987.

2] S. Das and S. P. Khatri, "Generation of the optimal bit-width topology of the fast hybrid adder in a parallel multiplier," in Proc. IEEE Int. Conf. Integr. Circuit Des. Technol., pp. 1–6, May 2009.

3] L. Ciminiera and P. Montuschi, "Carry-save multiplication schemes without final addition," IEEE Trans. Comput., vol. 45, no. 9, pp. 1050–1055, 2002.

4] M. C. Wen, S. J. Wang, and Y. N. Lin, "Low power parallel multiplier with column bypassing," in Proc. IEEE Int. Symp. Circuits Syst., pp. 1638–1641, 2010.

[15] N. Takagi and T. Horiyama, "A high-speed reduced-size adder under left-to-right input arrival," IEEE Trans. Comput., vol. 48, no. 1, pp. 76–80,Jan.2000.

[16] V. G. Oklobdzija, D. Villeger, and S. S. Liu, "A method for speed optimized partial product reduction and generation of fast parallel multipliers using an algorithmic approach," IEEE Trans. Comput., vol. 45, no. 3, pp. 294–306, Mar. 2012.

[17] A. Vazquez and E. Antelo, "Area and delay evaluation model for CMOS circuits," Internal Report, Univ. Santiago de Compostela (Spain), Jun. 2012.

[18] M. L. Hsia and O. T. C. Chen, "Low-power multiplier optimized by partial-product summation and adder cells," in Proc. IEEE Int. Conf. Circuits Syst., pp. 3042–3045, 2011.