Area Delay and Power Efficient 8-bit Booth Multiplier Design using Pre-Encoded 4-bit Radix Design

Sandeep Kumar Soni[1], Neelesh Gupta[2], Rajesh Sharma[3]

Truba College of Science & Technology, Bhopal, India

Abstract - In modern world of technology we focus on ICs design with low power techniques and also on reduction of area of an IC. In digital circuits the multiplication is very common operation which mostly used in DSP applications and in microprocessors. A multiplier using the radix-4 (or modified Booth) algorithm is very efficient due to the ease of partial product generation, whereas the radix-8 Booth multiplier is slow due to the circuit complexity and more numbers of partial products. In this paper we propose an approximate design for enhancing the system performance. At low radix the performance of system is better and power consumption of system is quite low but as we increase the radix the performance is poor as compare to lower one to overcome this a technique is proposed which improve the performance and power consumption on conventional deign. To implement such type of design we make an analysis on different adder architecture and compare their results then select best of the them. As various Booth multiplication techniques are available which design the radix-8 multiplier by using radix-4 technique. In our work we make an technique which provide us a power efficient, small area and fast operating Booth multiplier design.

Keywords: CLA, MBE, NR4SD, VHDL, Wallace Tree, CSA, RCA.

I. INTRODUCTION

Day by day IC technology is getting more complex in terms of design and its performance analysis. A faster design with lower power consumption and smaller area is implicit to the modern electronic designs. Multipliers generally have extended latency, huge area and consume substantial amount of power. Hence low-power multiplier factor style has become a very important region in VLSI system style. Everyday new approaches are being developed to style low-power multipliers at technological, physical, circuit and logic levels. Since the multiplier is usually the slowest component in an exceedingly system, the system's performance is decided by performance of the multiplier. Also multipliers are the most area consuming entity in a design. Therefore, optimizing speed and area of a multiplier is a major design issue nowadays. However, power and area of design are correlated with each other as increasing the speed results in larger in area and vice versa. So a compromise has to be done in speed of the circuit for a greater improvement in reduction of area and power.

A higher representation radix effectively indicates to fewer digits. Thus, a single-digit multiplication formula necessitates fewer cycles as we have a tendency to begin moving to a lot of higher radices that mechanically results in a lesser range of partial product. Thousands of algorithms are available for Many algorithms are developed for this purpose like Booth's formula, Wallace Tree technique etc. At the last step the addition is to be performed by using any of available adder like RCA, CLA or carry save adder (CSA). But to reduce the power consumption the summation architecture of the multiplier should be carefully chosen.

II. SYSTEM MODEL

Multiplying a variable by a group of identified constant coefficients may be a common operation in several digital signal process (DSP) algorithms. Compared to alternative common operations in DSP algorithms, like addition, subtraction, exploiting delay components, etc., multiplication is usually the foremost costly. There is a trade-off between the amount of logic resources used (i.e. the amount of silicon in the integrated circuit) and how fast the computation can be done. Compared to most of the other operations, multiplication requires more time given the same amount of logic resources and it requires more logic resources under the constraint that each operation must be completed within the same amount of time.

A general multiplier is needed if one performs multiplication between two arbitrary variables. However, when multiplying by a known constant, we can exploit the properties of binary multiplication in order to obtain a less expensive logic circuit that is functionally equivalent to simply affirming the constant on one input of a standard multiplier. In many cases, using a cheaper implementation for only multiplication still results in significant savings when considering the entire logic circuit because multiplication is relatively expensive. Furthermore, multiplication might be the vital operation, based on the application present.

A) Basic binary multiplier

The operation of multiplication is rather simple in digital electronics.

```
        10×8 = 80                      -6×4 = -24

          1 0 1 0                        1 0 1 0
          1 0 0 0                        0 1 0 0
  ----------------------------------------------------
        0 0 0 0                          0 0 0 0
      0 0 0 0                          0 0 0 0
    0 0 0 0                        1 1 1 0 1 0
  1 0 1 0                          0 0 0 0 0
  ----------------------------------------------------
  1 0 1 0 0 0 0                    1 1 1 0 1 0 0 0
```

It has its origin from the classical algorithm for the product of two binary numbers. This algorithm uses addition and shift left operations to calculate the product of two numbers. In booth multiplication algorithm two signed bit is multiplied using 2’s compliment notation. The rule was fabricated by Andrew Donald Booth in 1950 during doing analysis on crystallography at Birkbeck faculty in Bloomsbury, London. Booth used table calculators that were quicker at shifting than adding and created the rule to extend their speed.

B) A Typical Implementation

Booth's algorithmic program is enforced by repeatedly adding (with standard unsigned binary addition) one in every of 2 preset values A and S to a product P, then employing a rightward arithmetic shift on P. Let m and r be the number and multiplier, respectively; and let x and y represent the quantity of bits in m and r.

1) Determine the values of A and S, and the initial value of P. All of these numbers should have a length equal to (x + y + 1).

   a) A: Fill the most significant (leftmost) bits with the value of m. Fill the remaining (y + 1) bits with zeros.

   b) S: Far most vital bit is filled by 2’s complement notation and all remaining bits (y+1) with zero. c) P: Fill the foremost vital x bits with zeros. To the proper of this, append the worth of r. Fill the smallest amount vital (rightmost) bit with a zero.

2) Verify the 2 least vital (rightmost) bits of P.

   a) If they're 01, notice the worth of P + A. Ignore any overflow.

   b) If they're 10, notice the worth of P + S. Ignore any overflow.

   c) If they're 00, do nothing. Use P directly in the next step.

   d) If they are 11, do nothing. Use P directly in the next step.

3) Arithmetically shift the value obtained in the 2nd step by a single place to the right. Let P now equal this new value.

4) Repetition of step second and third is done until it done up to y times.

5) In this step the rightmost bit from P is dropped. This is often the product of m and r.

The representations of the number and products don't seem to be specified; usually, these are both additionally in two's complement illustration, just like the multiplier factor, however any mathematical notation that supports addition and subtraction can work still. As explicit here, the order of the steps isn't determined. It provide from least to most significant bit (MSB) with starting from i=0; hence multiplication of 2i is replaced in place of shifting of accumulator to the right between steps; low bits are often shifted out, and consequent additions and subtractions will then be done simply on the greatest N bits.

III. PREVIOUS WORK

[1] K. Tsoumanis and K. Pekmestzi, et al. In this work, proposes pre encoded radix-4 multiplier based on on-off encoding with extend to non redundant radix-4 signed digit (NR4SD) encoding. To this extend, the Non-Redundant radix-4 Signed-Digit (NR4SD) encoding technique, which uses the digit values {-1, 0,+1,+2 } or {-2,-1,0,+1}, is proposed leading to a multiplier design with less complex partial products implementation. Experimental results shows that the NR4SD multiplier technique is better on conventional booth multiplier as it is power efficient as well as in the coefficients memory.

[2] Dinesh, V and M. Kathirvelu et al. Multipliers are key parts of the many high performance systems like FIR filters, microprocessors, digital signal processors, etc. A multiplier is the slowest element and consumes an amount of power in system and also decides the performance of any system. The analysis of performance parameters of different multiplier logics is essential for design of a system intended for a specific function with constraints on Power, Area and Delay. The work presents a detailed analysis of all the serial-parallel and parallel architectures. The multipliers are designed for 4 bit multiplication using DSCH tool and the corresponding layouts are obtained using Micro wind 3.5 tool using 45nm technology. From the analysis it's discovered that the array multipliers give a general routing structure which can be optimum for FPGA based mostly systems. Among the tree based mostly multipliers Dadda multipliers have a small advantage over Wallace tree multipliers in terms of performance. The altered booth multiplier factor is relatively inefficient for bits lesser than or up to four, because of the enhanced space concerned for realization of the booth encoder and booth selector blocks. The analysis shows that for lower order bits Dadda reduction is the most efficient.

[3] N. G. NikDaud and M. S. Badruddin et al. One of the effective ways to speed up multiplication is by reducing the number of partial products and accelerating the accumulation. In this work, a new architecture of hybrid Modified Booth Encoded Algorithm (MBE) and Carry Save Adder (CSA) is developed as fast multiplier architecture. Altera Quartus II platform is used to run the simulation. The architecture design is programmed into FPGA using Altera DE2 board to verify the synthesizability on physical hardware. This hybrid fast multiplier delivers good performance in term of higher speed as well as in term of less usage of logic elements.

[4] S. Nithya and M. N. V. Nithya et al. They implement a high speed and low power FIR digital filter style employing the constant breadth booth multiplier. To scale back the miscalculation in constant breadth multiplier reconciling probability calculator is employed (ACPE). To realize higher speed, the altered Booth encryption has been used and conjointly to fast up the addition the carry look ahead adder is employed as a carry propagate adder. The multiplier circuit is intended employing VERILOG and synthesized exploitation Xilinx ISE9.2i machine. The area, power and delay of the designed filter is analyzed exploitation cadence tool.

[5] C. Xiaoping and W. Shumin et al. The radix-4 Booth encryption or altered Booth encryption (MBE) has been very much adopted in partial merchandise generator to style high-speed redundant binary (RB) multipliers. Attributable to the existence of the error-correcting word (ECW) generated by MBE and RB encryption, the RB multiplier generates a further RB partial product rows. An additional RB partial product accumulator (RBPPA) stage is required for 2n-b RB MBE multiplier. The upper radix Booth formula than radix-4 is adopted to scale back the amount of partial merchandise. However, the Booth encryption isn't economical due to the problem in generating laborious multiples. The laborious multiples drawback in RB multiplier is resolved by distinction of 2 easy power-of-two multiples. This work presents a brand new radix-16 RB Booth encryption (RBBE-4) to avoid the laborious multiple of high-radix Booth encryption while not acquisition any ECW. The planned technique results in build high-speed and low-power RB multipliers. The experimental results show that the planned RBBE-4 multiplier achieves vital improvement in delay and power consumption compared with the RB MBE multiplier and also the current accorded best RBBE-4 multipliers.

[6] R. P. Rajput and M. N. S. Swamy et al. This work presents the planning and implementation of signed-unsigned altered Booth encryption (SUMBE) multiplier factor. This altered Booth encryption (MBE) multiplier and therefore the Baugh-Wooley multiplier performs multiplication operation on signed numbers solely. The array multiplier and Braun array multipliers perform multiplication operation on unsigned numbers solely. Thus, the necessity of the new automatic data processing system may be a dedicated and really high speed novel multiplier unit for signed and unsigned numbers. Therefore, this work presents the planning and implementation of SUMBE multiplier. The altered Booth Encoder circuit generates half the partial merchandise in parallel. By extending sign bit of the operands and generating an extra partial product the SUMBE multiplier is obtained. The Carry Save Adder (CSA) tree and therefore the final Carry Look ahead (CLA) adder accustomed speed up the multiplier operation. Since signed and unsigned multiplication operation is performed by constant multiplier unit the desired hardware and therefore the chip space reduces and this successively reduces power dissipation and price of a system.

[7] B. C. Paul and M. Okajima et al. They tend to provide a ROM-based sixteen times sixteen multiplier for low-power applications. the planning uses sixteen four times four ROM-based multiplier blocks followed by carry-save adders and a final carry-select adder (all ROM-based) to get the thirty two bit output. All memory blocks are enforced employing single semiconductor memory cells and eliminating identical rows and columns for optimizing the facility and performance. Measuring ends up in 0.18 mum CMOS method show a 400th reduction in power over the standard carry-save array multiplier once operated at its highest frequency. The ROM-based style additionally provides four hundred and forty yards less delay than the array multiplier with a minimal increase (7.7%) in power. This demonstrates the low-power operation of the ROM-based multiplier additionally at higher frequencies.

IV. PROBLEM STATEMENT

One of the numerous solutions of realizing high speed multipliers is enhancing similarity that helps in decreasing the amount of consequent calculation levels. The initial version of Booth algorithmic program (Radix-4) had 2 explicit drawbacks. They were:

- The range of add-subtract operations and shift operations become variable and causes inconvenience in realizing parallel multipliers.
- The rule becomes inefficient once there are isolated 1’s.

These issues are overpowered by employing altered Radix4 Booth algorithmic program that scans strings of 3 bits employing the algorithmic program given below:

1. Lengthen the sign bit 1 position if necessary to ensure that n is even.
2. Add a 0 to the right of the LSB of the multiplier.

Corresponding to the value of each vector, each Partial Product will be 0, +M, -M, +2M or -2M.

V. PROPOSED METHODOLOGY

We used XILINX 14.1 platform to write our programs. All the RTL simulations have been done using this software only. Also for delay report the synthesis tool embedded in Xilinx was used. It takes HDL designs and synthesizes them to gate-level net-lists. Also it supports both Verilog and VHDL.

The basic building block for multiplication circuit that consist of four process module i.e. booth encoding, partial product summation, final addition and their accumulation shown in the figure 1.

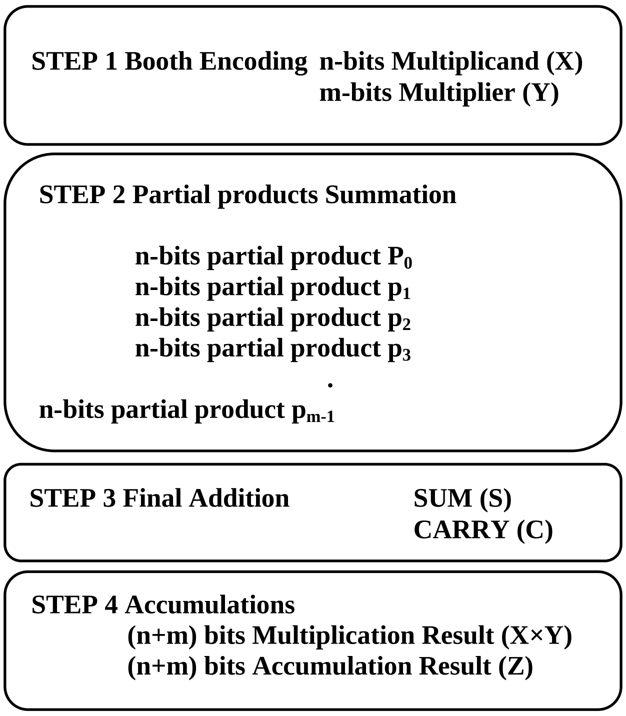

STEP 1 Booth Encoding
n-bits Multiplicand (X)
m-bits Multiplier (Y)
STEP 2 Partial products Summation
n-bits partial product P_0
n-bits partial product p_1
n-bits partial product p_2
n-bits partial product p_3
.
n-bits partial product p_{m-1}
STEP 3 Final Addition
SUM (S)
CARRY (C)
STEP 4 Accumulations
(n+m) bits Multiplication Result (X×Y)
(n+m) bits Accumulation Result (Z)

Figure 5.1 Basic Building Blocks for Multiplication Circuit

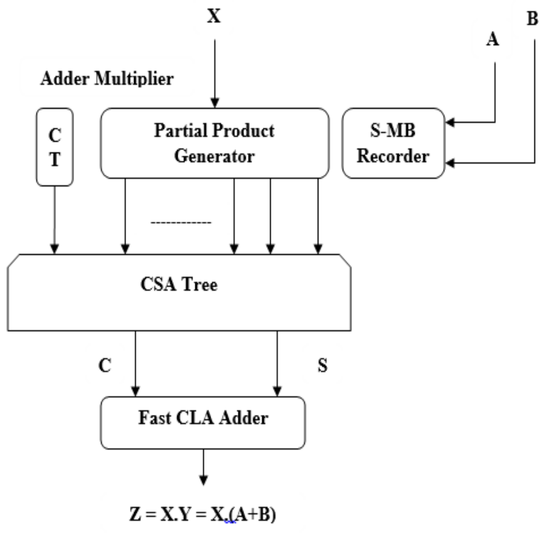


Figure 5.2 Fused design with direct recoding of the sum of A and B in its MB representation

In booth encoding define the number of bits of multiplicands and number of bits of multiplier and then partial product has been done. The summation of these partial products has done in very next process and then summation of these partial products has been done in process of final addition process and at the last result will store in the accumulator.

The step by step process of basic building block for multiplication circuit shown in the figure 2.

Step 1: Process of booth encoding in which n-bits of multiplicand (X) and m-bits of multiplier (Y) are involved. Product has done in this process.

Step 2: Process of partial product summation, in this process partial product summation has done by in shifting sequence.

Step 3: Process of final addition, in this process addition has done for all sequences which have generated through partial product summation process.

Step 4: Process of accumulation, in this process shows the results of multiplication and store into the memory.

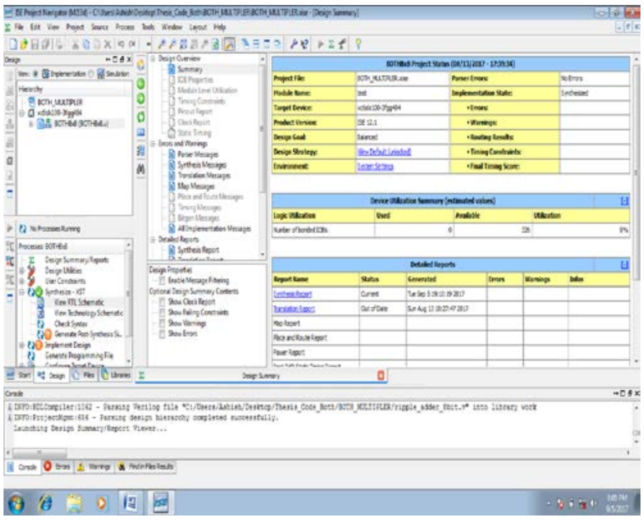
These steps of basic building block for multiplication circuit describe that the basic process multiplication.

VI. SIMULATION RESULTS

In this section discuss about the plan and reproduction of proposed model of 8 bit both multiplier using pre encoded 4-bit radix of design for the handling of diminishment of power utilization. The outline information driven mimic in Xilinx programming. The Xilinx programming version is 14.1.

The numerous sections of the proposed designed has been simulated and the separate results has been shown on GUI.

Booth Multiplier Experimetal Analysis:



```
INFO:HDLCompiler:1142 - Parsing Verilog file "C:/Users/Ashish/Desktop/Thesis_Code_Both/BOTH_MULTIPLIER/ripple_adder_8bit.v" into library work
INFO:ProjectMgmt:656 - Parsing design hierarchy completed successfully.
Launching Design Summary/Report Viewer...
```

Figure 6.1 Xilinx start for implementation.

In above figure when xiling window start for implementation and our implementation and product file and product version and booth proposed status..

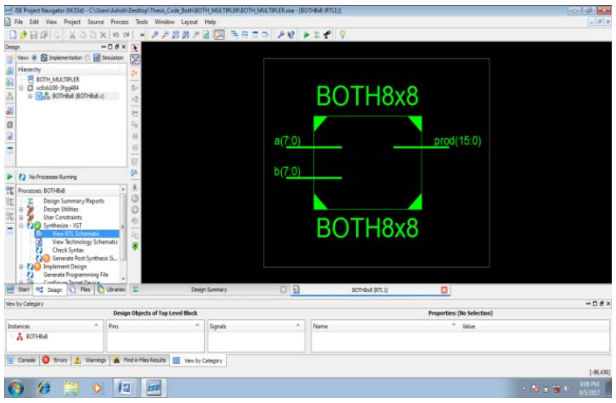


Figure 6.2 RTL schematic view

Figure show that the RTL schematics view in our implementation after the run the program.

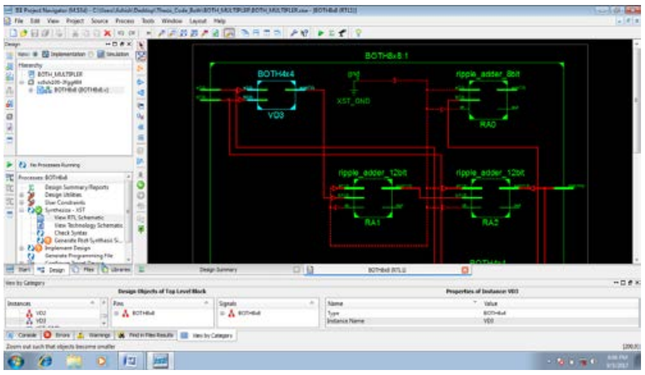
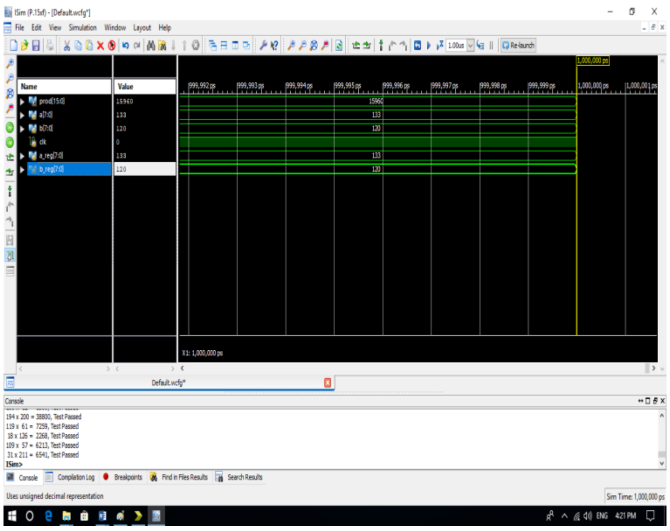


Figure 6.3 RTL schematic view with BOOTH combination.

Figure show that the RTL schematic view with zoom in our implementation after the run the program which shows that Booth 8×8 is combination of Booth 4×4.



```
194 x 200 = 38800, Test Passed
119 x  61 =  7259, Test Passed
 18 x 126 =  2268, Test Passed
109 x  57 =  6213, Test Passed
 31 x 211 =  6541, Test Passed
ISim>
```

Uses unsigned decimal representation

Figure 6.4 Test-bench Waveform generation using Xilinx-I-sim

Comparative Result Analysis

Table 1 mainly based on Number of 4 Input LUTS. DFD (%), RDFD (%), Utilization value of used and available parameter for the experimental process in a simulation model.

TABLE 1. Shows that the Data Driven input variance to the clock edge data-1110.

LOCAL UTILIZATION	Number Of 4 Input LUTS	Number of Occupied Files	Number of Slices Containing Only Related Logic	Number of Slices Containing Only Unrelated Logic	Number of Bonded
DFD (%)	17	3	97	2	11
RDFD (%)	20	5	98	4	14
USED	268	210	210	0	13
AVAILABLE	1536	768	210	210	124
UTILIZATION (%)	17	27	100	0	10

TABLE 2. Shows that the Data Driven input variance to the clock edge data-1101.

LOCAL UTILIZATION	Number of 6 Input LUTs	Number of Occupied files	Number of Slices Containing only related logic	Number of Slices Containing only unrelated logic	Number of Bonded
DFD (%)	17	31	98	1	14
RDFD (%)	19	34	100	3	17
USED	324	260	260	0	19
Available	1864	838	260	260	138
Utilization (%)	21	33	100	0	16

Above table 2 mainly based on Number of 6 Input LUTS.DFD (%), RDFD (%), Utilization value of used and available parameter for the experimental process in a simulation model.

TABLE 3. Shows that the resultant of delay, area and power using NR4SD-, NR4SD+ and Booth method.

Parameter / Method	Delay	Area	Power
NR4SD-	1.95	18357	11.00
NR4SD+	1.95	18485	11.10
PROPOSED BOOTH	1.70	18311	10.90

Table 4. Shows that the resultant of delay, area and power using NR4SD-, NR4SD+ and Booth method.

Parameter / Method	Delay	Area	Power
NR4SD-	1.98	19556	13.05
NR4SD+	1.98	19772	12.11
PROPOSED BOOTH	1.80	19300	11.05

Performance Result Evaluation:

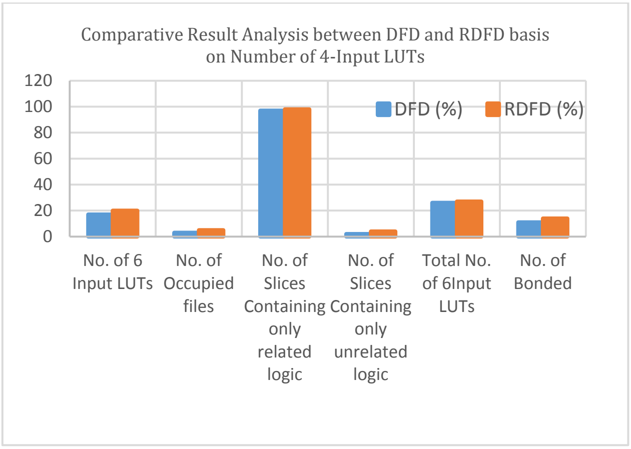


Figure 6.5 Comprative analysis of Used, Available and Utilization in logic(Number of 4 Input LUTs)

Above figure shows comparative analysis of different parameters on the basis of Number of 4-Input LUTs and clock edge data (1110) for DFD, RDFD, Used, Available and Utilization in Logic Circuit.

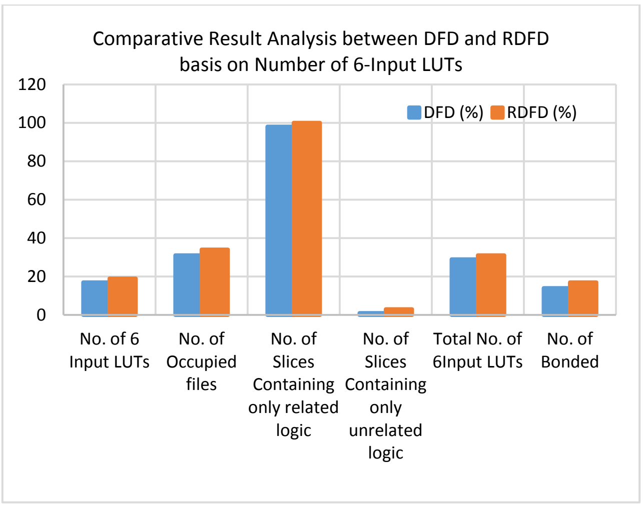


Figure 6.6 Comprative analysis of Used, Available and Utilization in logic(Number of 6 Input LUTs)

Above figure shows comparative result analysis of different parameters on the basis of Number of 6-Input LUTs and clock edge data (1101) for DFD, RDFD, Used, Available and Utilization in Logic Circuit.

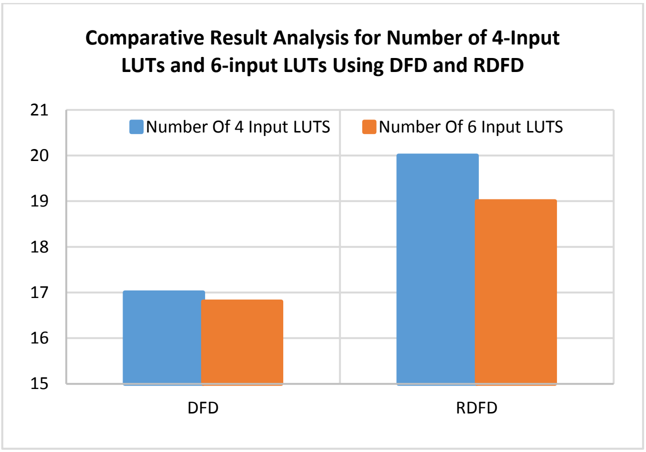


Figure 6.7 Comprative result analysis between DFD and RDFD for Number of 4-Input LUTs and Number of 6-Input LUTs in Logic Circuit.

VII. CONCLUSION

In our work we proposed a pre encoded multiplier design with off-line encoding and storing the coefficients in memory. We propose encoding these coefficients in the radix-8 booth multiplier using Non-Redundant radix-4 form. The proposed radix-8 booth multiplier using Non-Redundant radix-4 multiplier designs are more area and power efficient compared to the conventional and pre-encoded MB designs. Extensive experimental analysis verifies the gains of the proposed radix-8 modified booth multipliers in terms of area complexity and power consumption compared to the conventional MB multiplier.

VIII. FUTURE SCOPES

In our proposed work for future scope we can done on power estimation section and also on higher radix analysis. For further power analysis done on the gate-level by generating gate-level netlist and further the post layout analysis of this design can be done. In proposed work the estimated power is low when we compare with existing BM technique. For higher radix design can also make an analysis by making some improvements.

REFERENCES

[1] K. Tsoumanis, N. Axelos, N. Moschopoulos, G. Zervakis and K. Pekmestzi, "Pre-Encoded Multipliers Based on Non-Redundant Radix-4 Signed-Digit Encoding," in IEEE Transactions on Computers, vol. 65, no. 2, pp. 670-676, Feb. 1 2016.

[2] B. Dinesh, V. Venkateshwaran, P. Kavinmalar and M. Kathirvelu, "Comparison of regular and tree based multiplier architectures with modified booth encoding for 4 bits on layout level using 45nm technology," 2014 International Conference on Green Computing

Communication and Electrical Engineering (ICGCCEE), Coimbatore, 2014, pp. 1-6.

[3] N. G. NikDaud, F. R. Hashim, M. Mustapha and M. S. Badruddin, "Hybrid modified booth encoded algorithm-carry save adder fast multiplier," The 5th International Conference on Information and Communication Technology for The Muslim World (ICT4M), Kuching, 2014, pp. 1-6.

[4] S. Nithya and M. N. V. Nithya, "An efficient fixed width multiplier for digital filter," 2014 IEEE 8th International Conference on Intelligent Systems and Control (ISCO), Coimbatore, 2014, pp. 96-102.

[5] C. Xiaoping, H. Wei, C. Xin and W. Shumin, "A New Redundant Binary Partial Product Generator for Fast 2n-Bit Multiplier Design," 2014 IEEE 17th International Conference on Computational Science and Engineering, Chengdu, 2014, pp. 840-844.

[6] R. P. Rajput and M. N. S. Swamy, "High Speed Modified Booth Encoder Multiplier for Signed and Unsigned Numbers," 2012 UKSim 14th International Conference on Computer Modeling and Simulation, Cambridge, 2012, pp. 649-654.

[7] B. C. Paul, S. Fujita and M. Okajima, "ROM-Based Logic (RBL) Design: A Low-Power 16 Bit Multiplier," in IEEE Journal of Solid-State Circuits, vol. 44, no. 11, pp. 2935-2942, Nov. 2009.

[8] G. W. Reitwiesner, "Binary arithmetic," Advances in Computers, vol. 1, pp. 231-308, 1960.

[9] K. K. Parhi, VLSI Digital Signal Processing Systems: Design and Implementation. John Wiley & Sons, 2007.

[10] K. Yong-Eun, C. Kyung-Ju, J.-G. Chung, and X. Huang, "Csd- based programmable multiplier design for predetermined coefficient groups," IEICE Trans. Fundam. Electro. Comm. Comp. Sci.,vol.93,no.1, pp. 324-326, 2010.