ISSN 2349-7815

International Journal of Recent Research in Electrical and Electronics Engineering (IJRREEE)
Vol. 4, Issue 1, pp: (1-11), Month: January - March 2017, Available at: www.paperpublications.org

Implementation of Radix-4 Booth Multiplier by VHDL

Prof. Sneha Singh[1], Prachi Singh[2]

[1]Head of Department of Electronics & Communication Engineering, JNCT, REWA.
[2]M.E. Student, Department of Electronics & Communication Engineering, JNCT, REWA.

Abstract: **Latest technological development in VLSI design permits more functions integrated in a single chip. Multipliers are crucially important building structures for advanced computing and as a part of digital processing system. These logic and arithmetic structures should have to be speedy as well as precise enough so that number of such circuits can be integrated along a single chip. Considering this there is advancement in IC fabrication and design is still going on. In VLSI circuit area, power and delay are the parameters which are considered as design parameters. However, there exists a trade-off amongst them for an optimal design. Multipliers have very crucial and important part in designing of microprocessors, multimedia system and digital signal processors etc. Almost 15% of total IC power is consumed by multiplication unit alone. So it becomes very important to have a well organized design in terms of performance, area and its processing speed of multipliers and same as for Booth multiplication algorithm which gives a fundamental platform for such improvements in the designing of high speed multipliers with great performance.**

Booth algorithm gives such an efficient encoding scheme of the bits through first steps of the multiplication process. This work is based on configurable logic for 16-bit Booth multiplier using Radix-2 and Radix-4 Method. Booth multiplier can be configured to perform multiplication on 16-bit operands. The multiplier will identify the range of the operands during configuration register. The configuration register can be configured through input ports. The multiplier has been synthesized using Xilinx 14.5 and in this simulation we have achieve minimum combinational delay. Modelsim is used for the simulation part in this work.

Keywords: **Radix, XPS, VHDL, Modelsim, IC fabrication, CBM, MAC, RTL, CIAF, CLA.**

1. INTRODUCTION

Arithmetic and logic operations play an important role in digital circuits. Addition, multiplication, exponensation are important fundamental function in arithmetic operations and have wide applications in the field of engineering. The demand for high speed processing has been increasing as a result of expanding computer and signal processing applications. Higher throughput arithmetic operations are important to achieve the desired performance in many real-time signal and image processing applications [1]. Among these arithmetic operations multiplication is the key of almost every digital circuit. It is used extensively in many VLSI systems such as communication system architectures and microprocessors. Multiplication-based operations such as Multiply and Accumulate (MAC) and inner product are among some of the frequently used Computation-Intensive Arithmetic Functions (CIAF) currently implemented in many Digital Signal Processing (DSP) applications such as Convolution, Fast Fourier Transform (FFT), filtering, in microprocessors in its arithmetic and logic unit and in graphics [2].

Digital multipliers are the most commonly used components in any digital circuit design. They are fast, reliable and efficient components that are utilized to implement any operation. Depending upon the arrangement of the components, there are different types of multipliers available each offering different advantages and having tradeoff in terms of speed,

circuit complexity, area and power consumption. Reducing the time delay and power consumption are very essential requirements for many applications [1][3].

Multipliers have become a basic building block in computations especially in digital signal processing. Multipliers not only take a significant part of time delay, area cost but also cause high power consumption. To improve the speed and power dissipation of the multipliers, many techniques and design methodologies have been proposed. Most of the designs are targeted at a specific technology and require redesign for a new process technology. As a result, it is necessary to develop computation-efficient multipliers suitable for portable multimedia and digital processing systems, which require flexible processing ability, lesser switching activity and short design cycle. The biggest challenge faced with use of simple and conventional multipliers for multimedia and DSP systems is that the multiplier coefficients are not constant. If it is constant, a general multiplier can be simplified to a network of shift, adders and subtractors to reduce power consumption [4]. However, this kind of simplified multiplier is inflexible which makes it to be unsuitable for multiplication operations with varying coefficients. To achieve improved processing ability, various techniques for reconfigurable multipliers that are capable of supporting multiple-precision multiplications have been developed. Advanced VLSI technology has given designer the freedom to integrate many complex components, which was not possible in the past. Various high speed multipliers have been proposed and realized [5-11]. Among these multiplication algorithms Booth's multiplication is showing better performance. In this dissertation, an attempt has been made to combine configuration and range detection technique to design configurable Booth multiplier (CBM) that supports single 4-bit, single 8-bit, single 12-bit or single 16-bit multiplication. This CBM depends upon the output of the range detection technique with highly simplified circuit.

2. RELATED WORK

In literature review text is written by someone to consider the current points of current knowledge including substantive findings, as well as theoretical and methodological contributions to a particular topic. Lot of works have been done on Booth multiplier and is going on .In the following section a brief description on multipliers and Booth multiplier is presented:

Multiplication in hardware can be implemented in two ways either by using more hardware for achieving fast execution or by using less hardware and end up with slow execution [12]. The area and speed of the multiplier is an important issue, increment in speed results in large area consumption and vice versa. Multipliers play vital role in most of the high performance systems. Performance of a system depends to a great extent on the performance of multiplier thus multipliers should be fast and consume less area and hardware. Booth's multiplication resulted in the reduction of the maximum height of the partial product array, which may simplify the partial product reduction tree, in terms of delay and regularity of the layout [12]. This is of special interest for short bit-width multipliers for high performance, where short but high speed bit-width multiplications are common operations.

Different adders are compared by using critical parameters like Delay, Power, and Area etc. to make clear ideas of which adder was best suited for situation. After comparing all, it was concluded that Carry Select Adders are best suited for situations where speed is the critical concern [13]. Coming to Multipliers ,implementation of Radix-2 Booth Multiplier is done using different adder and came to final conclusion that parallel multipliers are much better than the serial multipliers due to less area consumption and hence the less power consumption [13].

In many DSP applications, all of multiplier output bits were not used, but only upper bits of output were used. Kidamhi proposed truncated unsigned multiplier for this idea. This truncation scheme can be applied to Booth multiplier which can be used in real DSP systems more efficiently. Also, truncated Booth multiplier guaranteed 0 input to 0 output that was not provided before. Truncated Booth multiplier reduced about 37-48 % of area and about 44 % of power consumption [14].

Different design methods are proposed to develop a modular approach for optimizing power consumption [7]. It is found that algorithm based design reduce gate switching activity considerably and as result power consumption in multiplier is reduced [15]. It is found that data complexity and various combination of gate level digital circuit has considerable impact in power dissipation. Beside this physical design of the chip can be optimized by using Genetic Algorithm by analyzing placement option subject to optimum space allocation. Similarly selection of Booth Algorithm and Modified Booth

Algorithm may reduce power consumption as consequence of data complexity. It is found that in multiplier circuit, Modified Booth Algorithm reduces power consumption as compared to other methods of multiplication [15].

On making a comparison between radix 2 and radix 4 Booth multiplier in terms of power saving experimental results demonstrate that the modified radix 4 Booth multiplier has 22.9% power reduction than the conventional radix 2 Booth Multiplier [16].

Booth multiplier can be configured based on dynamic range detection of multipliers and optimized for low power and high speed operations and which can be configured either for single 16-bit multiplication operation, single 8-bit multiplication or twin parallel 8-bit multiplication is designed [17]. It was found that Booth Multiplier can efficiently deactivate ineffective circuitry which were not produced effective result so that speed of operation gets increases and device area is reduced so that power gets reduced.

From the above discussion it is clear that Booth multiplier makes the multiplication process simple, reduces the height of partial product generated. Also Booth multiplier implemented using carry select adders are best suited where speed is the critical concern. Modified Booth's algorithm reduces power consumption when compared to other multiplication algorithms**.** Research on power analysis was done to a greater extent, however from delay perspective little amount of contributions are made. Booth multiplier can be configured to perform multiplication only on significant bits and thereby making it delay efficient. So, an attempt has been made in this dissertation to make the Booth multiplier delay efficient. By doing this switching can be reduced thereby decreasing power consumption.

3. METHODOLOGY

Here an attempt is made to design a high speed and power-efficient configurable Booth Multiplier (CBM). Configurable Booth multiplier can be twice as fast as Booth's algorithm. CBM is an efficient way to reduce the number of partial products. The main concerns are speed, power efficiency and structural flexibility.

3.1 Configurable Booth Multiplier

From the basics of Booth multiplication algorithm we came to know that number of passes or cycles to obtain the final product depends upon the operand width. If the input data is of the form of 001110 (multiplier) and 001001(multiplicand) we have to go for six passes to obtain the result.Since the significant data is contained in least 4 significant bits, output result can be obtained only after four passes by suppressing most significant bits, being zero .That will not only reduce the delay but also reduces the switching to a greater extent. So, an attempt has been made to make the Booth multiplier configurable to reduce the delay and power.

In this multiplication technique of configurable booth multiplier, firstly the range of both the operands A and B are detected by the Configuration register that is being configured through input ports. Configuration register will detect whether the computation will be done on 4 bit, 8 bit, 12 bit or 16 bit. There after the further computation will be done accordingly.

A register PA is taken, that will store the concatenation of accumulator (4 bit or 8bit or 12 bit or 16 bit, initially assigned with 0's), multiplier A (4 bit or 8 bit or 12 bit or 16 bit) and an extra least significant bit, LSB (initially assigned with 0). PA [1:0] is checked whether it is 00, 01, 10, and 11. If PA [1:0] = 01, then PA + B operation will occur with single arithmetic right shift, and if PA [1:0] = 10 then PA – B will be calculated wit h single arithmetic right shift, else if PA[1:0] =00 or PA[1:0] = 11 then only single arithmetic right shift will be done on PA. This whole procedure of checking of PA [1:0] bits will be done repeatedly and the number of iterations will depend upon the range detected. Finally, the final output will be saved in register P.

Figure 3.1 describes the working of configurable booth multiplier. In this figure multiplier and multiplicand are stored in 16-bit registers. Configuration register detects the range of input data by performing bitwise OR operation .Booth controller provides the necessary control signals and addition, subtraction and shifting will be done according to Booth's algorithm. Final results are stored in accumulator.

ISSN 2349-7815

International Journal of Recent Research in Electrical and Electronics Engineering (IJRREEE)
Vol. 4, Issue 1, pp: (1-11), Month: January - March 2017, Available at: www.paperpublications.org

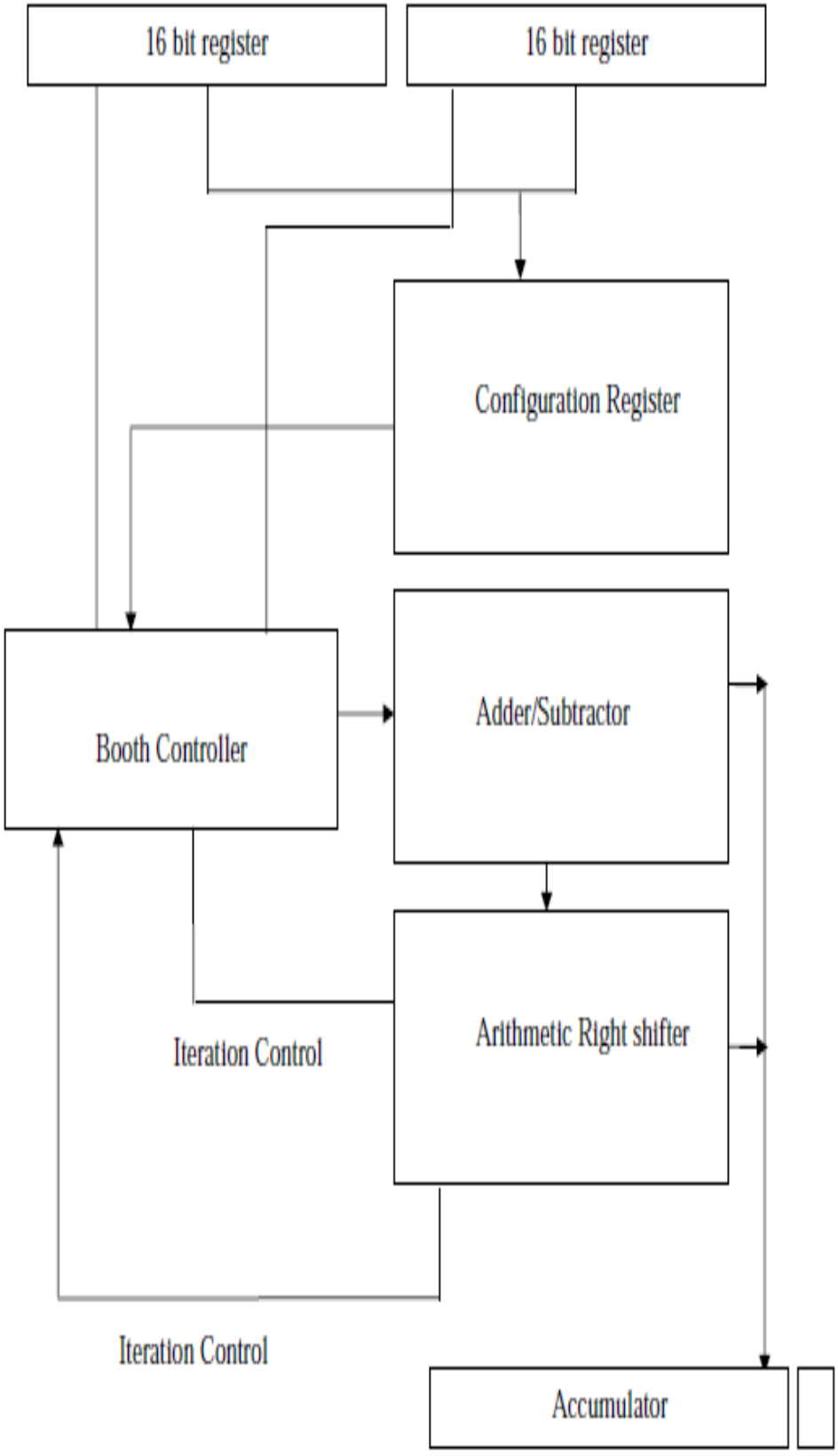

Figure 3.1: Working of Configurable Booth Multiplier

Figure 3.2 describes the operation of configurable booth multiplier.

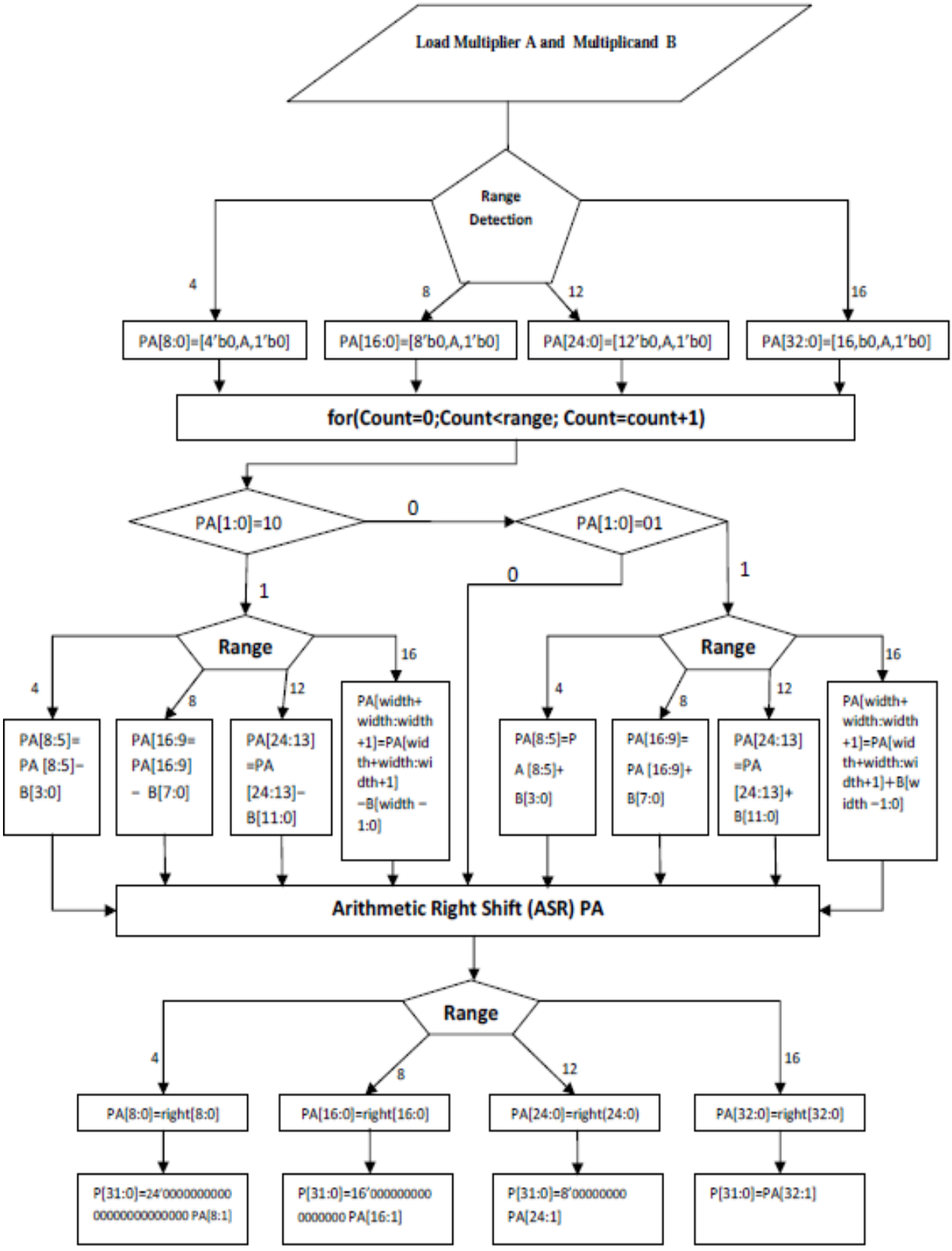

Figure 3.2: Booth Multiplier Flow Chart

In this figure after multiplier and multiplicand are loaded, the configuration register detects the range of the data. Range can be 8,4,12 or 16 depending upon the result of bitwise OR operation. The results are then stored in a register PA. Then after the further computation are done accordingly as per Booth's algorithm. By doing so, the calculations will be cut shorter to much extent. With another advantage of simplicity, this circuit is preferred so as to suppress the most significant bits of the operands for multiplication if they are all zero.

Booth Multiplier using Radix 4:

One of the solutions of realizing high speed multipliers is to enhance parallelism which helps to decrease the number of subsequent calculation stages. The original version of the Booth algorithm (Radix-2) had two drawbacks. They are:

(i) The number of add subtract operations and the number of shift operations becomes variable and becomes inconvenient in designing parallel multipliers.

(ii) The algorithm becomes inefficient when there are isolated 1's. These problems are overcome by using modified Radix 4.

Booth algorithm which scans strings of three bits is given below:

1) Extend the sign bit 1 position if necessary to ensure that n is even.

2) Append a 0 to the right of the LSB of the multiplier.

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

The negative values of B are made by taking the 2's complement and in this paper Carry-look-ahead (CLA) fast adders are used. The multiplication of M is done by shifting M by one bit to the left. Thus, in any case, in designing n-bit parallel multiplier, only n/2 partial products are produced.

The partial products are calculated according to the following rule

$$\mathbf{Z_n = -2 \times B_{n+1} + B_n + B_{n-1}}$$

Where, B is the multiplier.

Consider example for radix 4:

Multiplicand 1 0 0 0 0 0 0 1

Multiplier 0 1 1 1 1 1 1 0 0

+2 0 0 -2

0000000011111110

00000000000000

000000000000

1100000010

Product 1100000101111110

Table 3.2: Modified Radix 4 Recoding Rules

B	Zn	Partial Product
r000	0	0
001	1	1×Multiplicand
010	1	1×Multiplicand

011	2	2×Multiplicand
100	-2	-2×Multiplicand
101	-1	-1×Multiplicand
110	-1	-1×Multiplicand
111	0	0

4. RESULT AND DISCUSSION

4.1 Simulation result for Radix-4

In the design of Booth multiplier for radix 4 there are two inputs namely multiplier [7 :0] and multiplicand [7 : 0] and the single output Product [16:0]. Output is verified for inputs of different ranges. In this case the simulation results are shown for input value in binary for multiplier "00000010" and for multiplicand "00000010" and we will get the output in product is "0000000000000100" shown in fig 4.1:

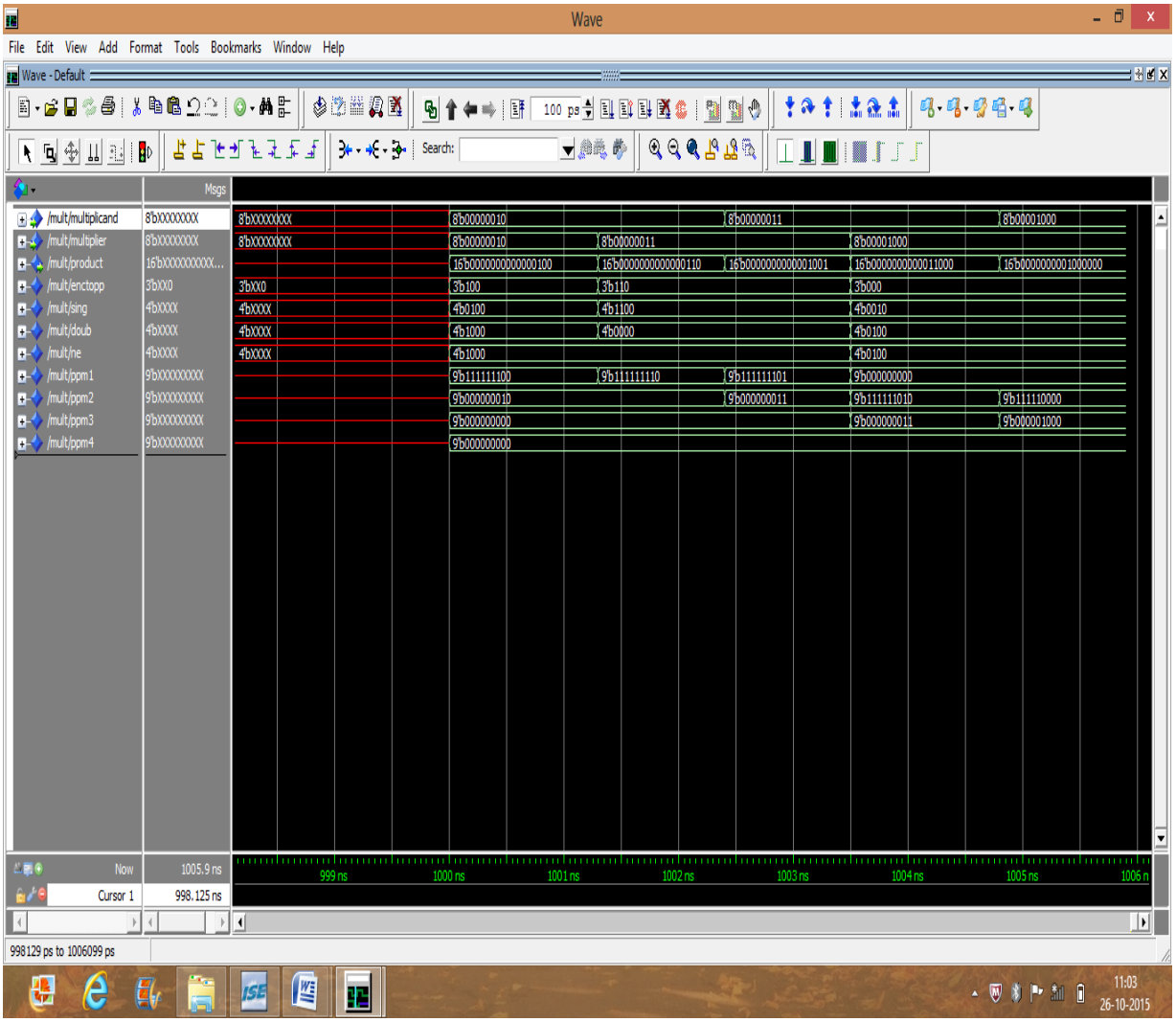

Fig 4.1: Simulation for case-1 using Radix-4

In the design of Booth multiplier for radix 4 there are two inputs namely multiplier [7 :0] and multiplicand [7 : 0] and the single output Product [16:0]. Output is verified for inputs of different ranges. In this case the simulation results are shown for input value in binary for multiplier "00000010" and for multiplicand "00000011" and we will get the output in product is "0000000000000110" shown in fig 4.2:

ISSN 2349-7815

International Journal of Recent Research in Electrical and Electronics Engineering (IJRREEE)
Vol. 4, Issue 1, pp: (1-11), Month: January - March 2017, Available at: www.paperpublications.org

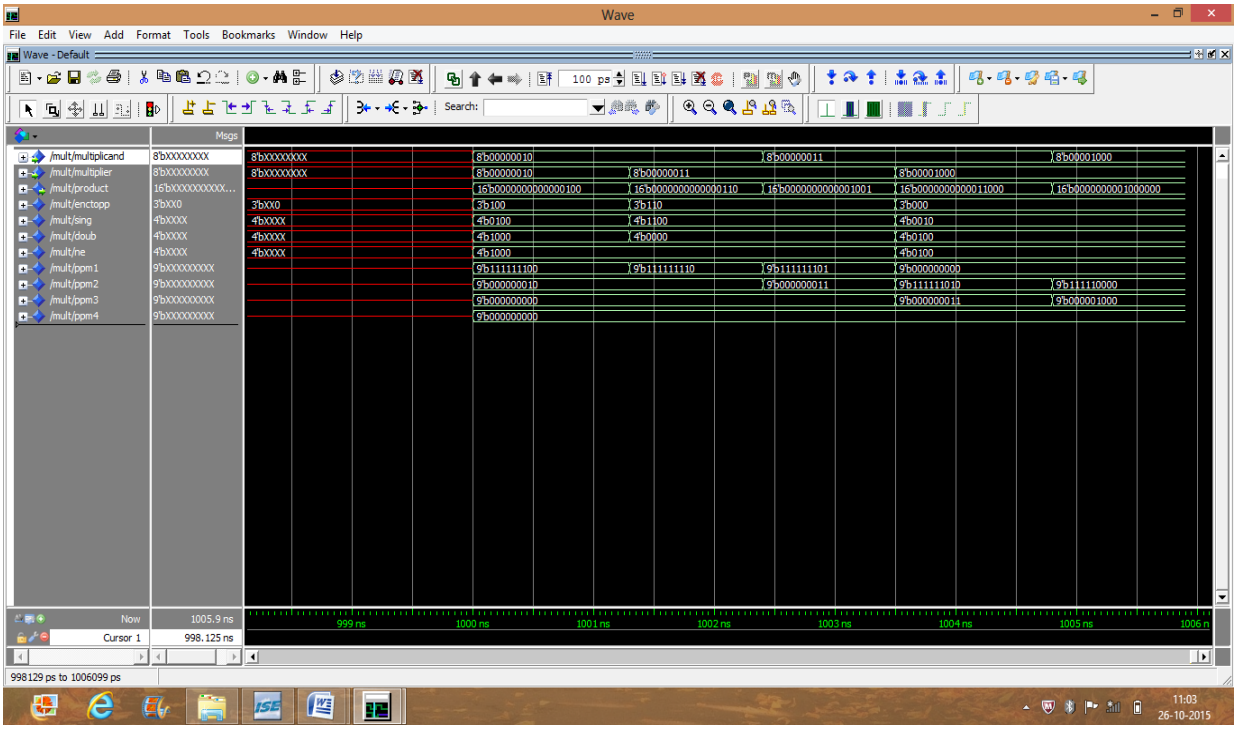

Fig 4.2: Simulation for case 2 using Radix 4

In the design of Booth multiplier for radix 4 there are two inputs namely multiplier [7 :0] and multiplicand [7 : 0] and the single output Product [16:0]. Output is verified for inputs of different ranges. In this case the simulation results are shown for input value in binary for multiplier "00000010" and for multiplicand "00000100" and we will get the output in product is "0000000000001000" shown in fig 4.3:

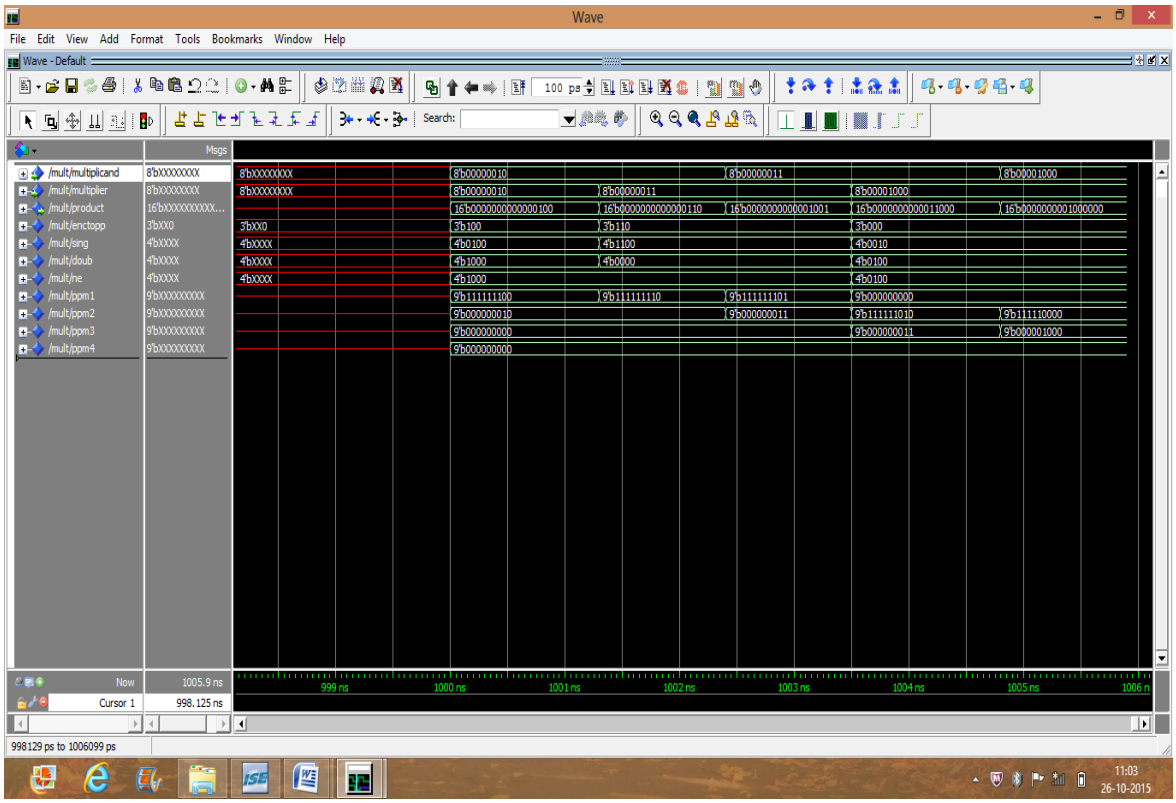

Fig 4.3: Simulation for case 3 using Radix 4

ISSN 2349-7815

International Journal of Recent Research in Electrical and Electronics Engineering (IJRREEE)
Vol. 4, Issue 1, pp: (1-11), Month: January - March 2017, Available at: www.paperpublications.org

4.4 Synthesis Result for Radix 4:

The multiplier has been synthesized using Xilinx ISE 8.1i. The RTL Schematic of Booth Multiplier using Radix 4 has been shown in figure 4.4 In the RTL schematic of Booth multiplier multiplicand [7:0] and Multiplier [7:0] represents the 8-bit input operands and Product (16:0) represents 16-bit output product, fig 4.5 shows the internal RTL of the Booth Multiplier and fig 4.6 shows the device utilization of Booth Multiplier using Radix 4.

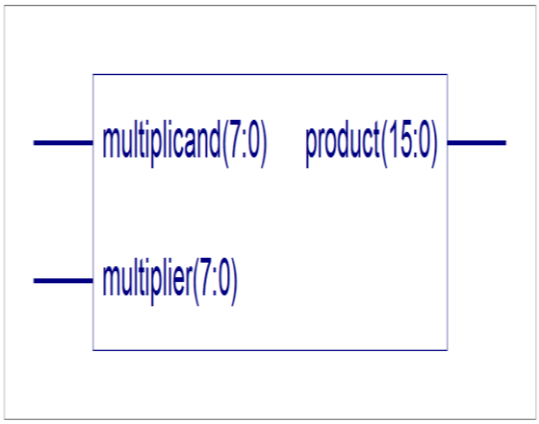

Fig 4.4: RTL of Booth Multiplier using Radix 4

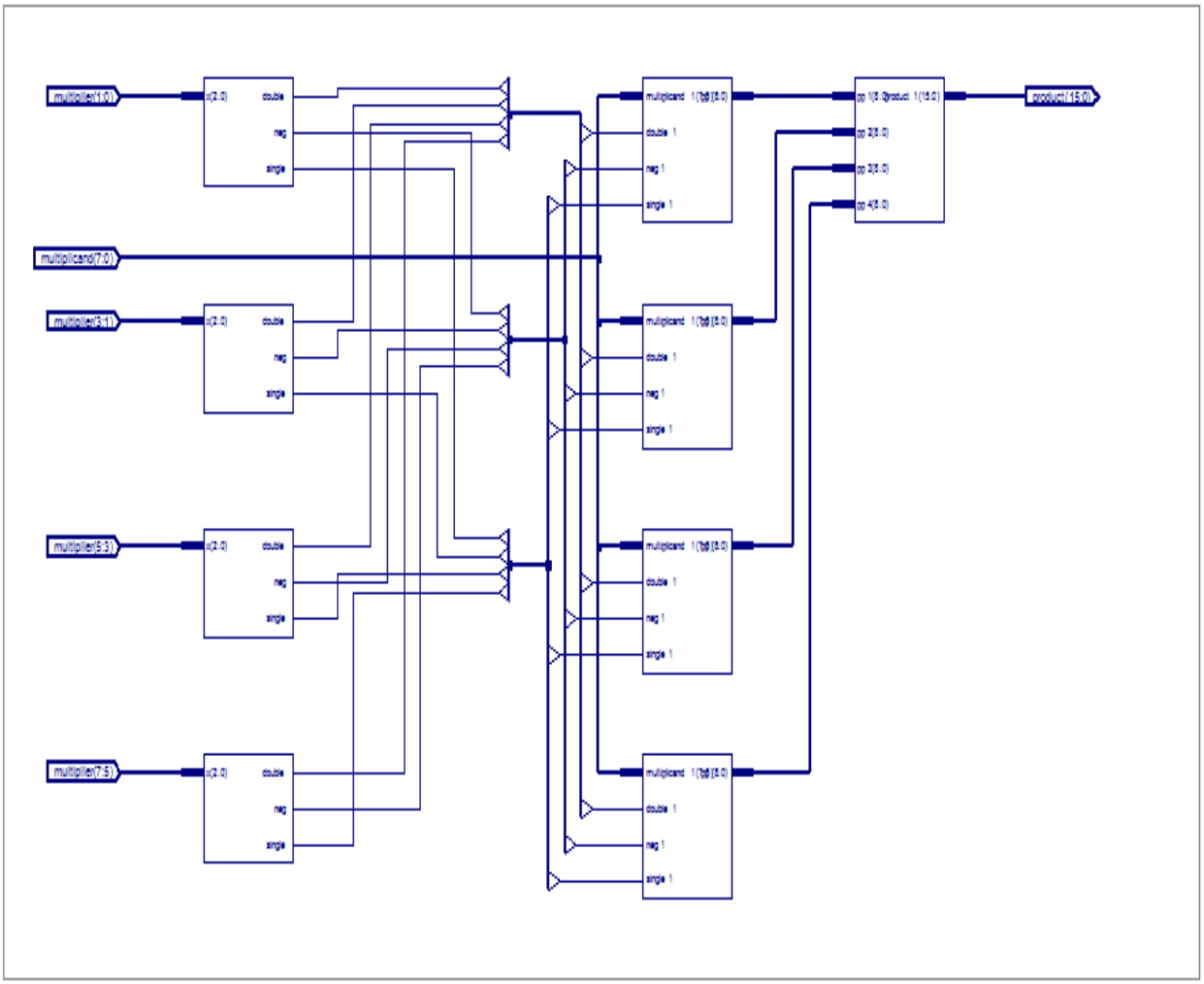

Fig 4.5: Internal RTL of Booth Multiplier using Radix 4

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	84	192	43%
Number of 4 input LUTs	155	384	40%
Number of bonded IOBs	32	90	35%

Fig 4.6: Device Utilization of Booth Multiplier using Radix 4

5. CONCLUSION

After going through all the hard work and facing problems, this project managed to complete its objectives that are to study the Booth Multiplier and design of Delay Efficient Booth Multiplier using Radix-2 and Radix-4 Method. The main aim of implementing the Booth multiplier is to carry out the fractional products to lessen the delay, increase the speed of operation and to lessen the power consumption in the circuit. We have presented a 16-bit Booth multiplier using Radix2 and Radix4 Method. This multiplier can be configured to perform 16 bit multiplication depending upon the output of configuration register. The multiplier will detect the range of the operands through configuration register. The configuration register can be configured through input ports. It also deactivate the unneeded switching activities in useless range possible as much. The output product of the multiplier can be abridged to further reduce power consumption by sacrificing a bit of output exactness. The multiplier is synthesized using Xilinx. The delay results are then compared with Booth multiplier using radix 2 and Booth Multiplier using Radix 4 and it was found that Booth multiplier using radix 4 gives a less delay of logic.

REFERENCES

[1] Himanshu Thapliyal and Hamid R. Arabnia, “A Time-Ar ea- Power Efficient Multiplier and Square Architecture Based On Ancient Indian Vedic Mathematics”, Department of Computer Science, The University of Georgia, 415 Graduate Studies Research Center Athens, Georgia 30602-7404, U.S.A ,2003.

[2] Purushottam D. Chidgupkar and Mangesh T. Karad, “Th e Implementation of Vedic Algorithms in Digital Signal Processing”, Global J. of Engng. Educ., Vol.8, UICEE Published in Australia. 2004.

[3] E. Abu-Shama, M. B. Maaz, M. A. Bayoumi, “A Fast an d Low Power Multiplier Architecture”, The Center for Advanced Computer Stu dies, The University of Southwestern Louisiana Lafayette, 2007.

[4] Shiann Rong Kuang and Jiun-Ping Wang “ Design of Po wer Efficient Configurable Booth Multiplier” Vol. 57, No.3, March 2010

[5] A. D. Booth, “A Signed Binary Multiplication Techni que”, Quarterly J. Mech. Appli. Math., vol 4, part2, pp. 236-240 , 1951

[6] O. L. Mac Sorley, "High Speed Arithmetic in Binary Computers", Proceedings of IRE, Vol.49 , No. 1, January, 1961

[7] D. Villeger and V. G. Oklobdzija, “Analysis Of Boo th Encoding Efficiency In Parallel Multipliers Using Compressor For Reduction Of Partial Products", Proceedings of the 27th Asilomar Conference on Signals Systems and Computers, pp. 781-784, 1993.

[8] R. S. Lim, “High Speed Multiplication and Multiple Summand Addition”, 4 th International Symposium on Computer Arithmetic, Santa Monica, California, June 1978.

[9] Hsin-Lei Lin, “Design of a Novel Radix – 4 Booth Mu ltiplier”, the 2004 IEEE Asia – Pacific Conference on Circuit and Systems, December 2005

[10] Wen-Chang & Chein-Wei “High-speed Booth Encoded Par allel Multiplier Design”, IEEE Transactions on Computer, 2000.

[11] E .A. Razaidi, " Analysis of various Modified Booth Encoder (MBE) and Proposal for an Efficient Modified Booth Encoder", IEEE Regional Sy mposium on Microelectronics, December, 2007

[12] Deepali Chandel, Gagan Kumawat , Pranay Lahoty, Vidhi Vart Chandrodaya , Shailendra Sharma "Booth Multiplier: Ease of multip lication" International Journal of Emerging Technology and Advanced Engineering Volume 3, Issue 3, March 2013

[13] Sakshi Rajput,Priya Sharma,Gitanjali and Garima " H igh Speed and Reduced Power – Radix-2 Booth Multiplier" International Journal of Computational Engineering & Management, Vol. 16, Issue 2, March 2013

[14] Kwang Hyun Lee , Chong Suck Rim, " A Hardware Reduc ed Multiplier for Low Power Design" Proceedings of the second IEEE Asia Pacific Conference,Korea, 2000.

[15] Zamin Ali Khan ,S. M. Aqil Burney , Jawed Naseem, Kashif Rizwan. "Optimization of Power Consumption in VLSI Circuit" International Jo urnal of Computer Science Issues, Vol. 8, Issue 2, March 2011

[16] Nishat Bano "VLSI Design of Low Power Booth Multipl ier" International Journal of Scientific & Engineering Research, Volume 3, Issue 2, February -2012

[17] J. Sreenivasulu Reddy , Y.Avanija , M. Mahesh Babu "Dy namic Range Detection Based Booth multiplier for Low Power and high Speed Applications" International Journal of Emerging trends in Engineering and Development Issue 2, Vol.6, September 2012.

[18] Parth Sarthy Mohanty ,"Design and Implementation of Low Power Fast Multipliers", thesis report NIT Rourkela, 2009.

[19] Padmanabhan Bala Subramanian and Nikos E. Mastorakis, "High Speed Gate Level Synchronous Full Adder Designs," Wseas Transcation s on Circuits and Systems ,Issue 2, Volume 8, pp 290-300, February 2009.

[20] Sanjiv Kumar Mangal, Rahul M. Badghare," FPGA Imple mentation of Low Power Parallel Multiplier",10 th International Conference on VLSI Design, 2007.

[21] Yingtao Jiang, Abdulkarim Al-Sheraidah, Yuke Wang, Edwin Sha, and Jin-Gyun Chung " A Novel Multiplexer-Based Low-Power Full Adder" in IEEE transactions on circuits and systems ,vol. 51, no. 7, July 2004

[22] Harpreet Singh Dhillon and Abhijit Mitra, "A Reduce d- Bit Multiplication Algorithm for Digital Arithmetic", International Journal of Compu tational and Mathematical Sciences , 2008.

[23] Fayed and M. A. Bayoumi, "A Novel Architecture for Low-Power Design of Parallel Multipliers," Proceedings of the IEEE Computer Soci ety Workshop on VLSI, pp.149-154, 2001.

[24] Earl E. Swartzlander, "Computer Arithmetic" Vol. 1 & 2, IEEE Computer Society Press, 1990.

[25] M. Ercegovac, "Digital Systems and Hardware/Firmwa re Algorithms", Chapter 12: Arithmetic Algorithms and Processors, John Wiley & Sons, 1985.