Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 09, September-2016

Design and Implementation of Radix 4 Based Multiplication on FPGA

Supriya S. Saste[1]
Dept. of Electronics & Telecommunication
Trinity College of Engineering & Research
Pune, India.

Prof. Anil G. Sawant[2]
Dept. of Electronics & Telecommunication
Trinity College of Engineering & Research
Pune, India.

***Abstract*— With the recent rapid increase in scale of integration, many sophisticated signal processing as well as video processing systems are being implemented on VLSI chip in which multiplication is dominant operation. The performance of these systems is based on computation capacity and power consumption. This paper presents novel approach of multiplication scheme based on Radix 4 and its implementation on FPGA which results in great computational capacity and reduced power consumption. This system has been designed and simulated using Xilinx 13.4 for 8x8 bit numbers.**

Key Words—Booth's Algorithm, Radix 4, VLSI, Xilinx 13.4.

I. INTRODUCTION

Multiplication is one of the most important arithmetic operations which is used in high performance systems such as microprocessors, digital signal processors and multimedia applications [1][2][5]. Previously multiplication was done by repetitive sequence of other two basic arithmetic operations viz., addition, subtraction along with shift operations. Hence, multiplication is repetitive addition of numbers. The 'multiplicand' is a number which is to be added and number of times it is added is called as 'multiplier'. The repetitive addition method to employ multiplication is comparatively slow.

Multiplication is mainly performed in three different stages: In first stage partial products are generated. The next stage i.e. stage two deals with reduction of partial products and finally in stage three all the partial products are summed together to get the final result of multiplication operation. The fundamental principle of multiplication is generation of partial products and accumulation of partial products [2]. Multiplication can be performed both on signed as well as unsigned numbers [3]. However signed multiplication is careful operation. Signed numbers cannot be multiplied in same manner as that of the unsigned numbers. Here the Booth's algorithm comes in.

The motivation of Booth's multiplication scheme is to increase the speed of multiplication process. As compared to conventional methods Booth's multiplication helps to reduce the number of iteration steps and results in faster computation. In this paper we present 8 bit multiplication by using modified Booth's (Radix 4) algorithm and its implementation on hardware platform.

II. BOOTH'S RECODING (RADIX 2) ALGORITHM

The Booth's algorithm was invented by Andrew D. Booth which employs multiplication of both signed and unsigned numbers. This algorithm has been used to generate the partial products which firstly encode the multiplier bits. Radix-2 and Radix-4 are two algorithms which generate reduced and efficient partial products for multiplication [3]. The basic technique stated by Booth is explained further.

The technique invented by Booth allows for smaller and faster multiplication of binary integers in 2's complement representation. In order to do multiplication by Booth's recoding algorithm, we have to recode the multiplier first. Each bit of the recoded multiplier can take any value from: 0, 1 and -1. In order to do this, 2 bits of multiplier are compared at a time by overlapping technique. Thus, in Radix-2 grouping of multiplier bits starts from LSB for which the first block uses only single bit of multiplier and another bit is assumed zero [4]. The recoded multiplier for Radix-2 is obtained by performing following steps:

i) Add a zero to the LSB side of given multiplier.

ii) By using overlapping technique, group two bits of multiplier and recode the number using following table:

TABLE I. RADIX-2 BOOTH ENCODING

Xn	Xn+1	Recoded Bits	Operations Performed
0	0	0	0
0	1	+1	1*Multiplicand
1	0	-1	-1*Multiplicand
1	1	0	0

Consider following example in which multiplicand and multiplier have 4 bits.

Multiplicand 1100
Multiplier 1010

So, according to the table shown above the recoding bits will be obtained as partial product:

PP0=00000
PP1=00100
PP2=11100
PP3=00100

Finally, all the partial products are added to get final product result.

The main version of Booth's algorithm (Radix-2) had two drawbacks:

1) With the invariability of add/subtract operations, the algorithm became inconvenient while designing parallel multipliers.

2) If there is a string of isolated 1s, the algorithm becomes inefficient [10].

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 09, September-2016

The drawbacks enlisted in Booth's algorithm are overcome by using Modified Booth's Algorithm (Radix-4).

III. MODIFIED BOOTH'S (RADIX 4) ALGORITHM

The number of bits multiplier/multiplicand is composed of, gives exact number of partial products generated in multiplication operation. So, to perform the addition of partial product is main bottleneck in multiplication operation and considered as the important factor to speed up multiplication. In Booth's recoding (Radix-2) algorithm, if 2 'n' bit numbers are multiplied then 'n' partial products will be generated. The desired high speed can be achieved if the partial products are reduced. Modified Booth's (Radix 4) Algorithm uses the technique of partial product reduction to speed up multiplication operation. So, if 2 even 'n' bit numbers are multiplied, the number of partial products generated is 'n/2' and if 'n' is odd, number of partial products are 'n+1/2'.Thus, in Radix 4 the number of partial products is reduced to half. To have high speed multipliers, Modified Booth's Algorithm is an ultimate solution. This algorithm scans strings of three bits at a time.

The numbers of steps involved in Radix 4 multiplication algorithm are shown below:

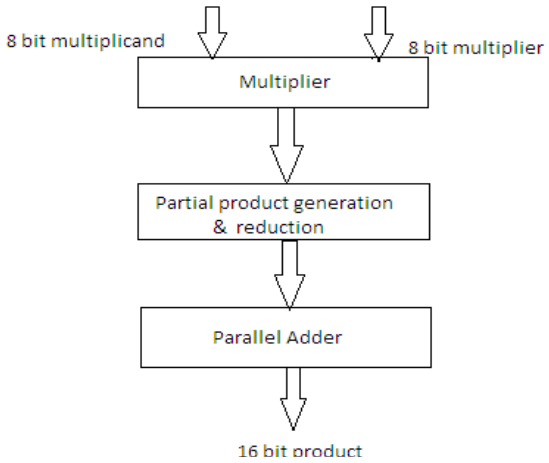

Fig.1 Radix 4 multiplication steps

In Modified Booth's (Radix 4) Algorithm, the multiplicand is recoded based on bits of multiplier which can take any value from + 1, + 2 or 0. Three bits of multiplier are compared at a time, by using overlapping technique. Similar to Radix 2, we have to group bits of multiplier starting from LSB for which first block only uses two bits, considering third bit as zero. Following steps are to be performed in order to generate recoded multiplier of Radix-4:

a) In order to ensure that n is even, extend the sign bit 1 position (if necessary).
b) Add a 0 to right of the LSB of multiplier.
c) Based on the value of each recoded bit, each partial product will be 0, +M, -M, +2M or -2M.

For Radix 4 bit pairing is done as shown below:

$\overline{0\ 1}\ \overline{1\ 0}\ \overline{0\ 1}\ 1\ 0$

Following table depicts the functional operation of Radix 4 Booth encoder:

TABLE II. RADIX-4 ENCODING RULES

Xn	Xn+1	Xn-1	Recoded Bits	Operations Performed
0	0	0	0	0
0	0	1	+1	+M
0	1	0	+1	+M
0	1	1	+2	+2M
1	0	0	-2	-2M
1	0	1	-1	-1M
1	1	0	-1	-1M
1	1	1	0	0

In above table 'M' is nothing but multiplicand

Consider multiplicand and multiplier is composed of 4 bits respectively.

Multiplicand 1100
Multiplier 1010

So, according to Radix 4 recoding rules, partial products obtained are:

PP0= 1000
PP1= 0100

From above example, it can be concluded that if there is 4 bit number, obtained partial products are 2 i.e. for 'n' bit number we get 'n/2' partial products. Since, the number of partial products are reduced, speed of multiplication process increases. Final product of multiplication is obtained by adding partial products.

IV. RESULTS AND DISCUSSION

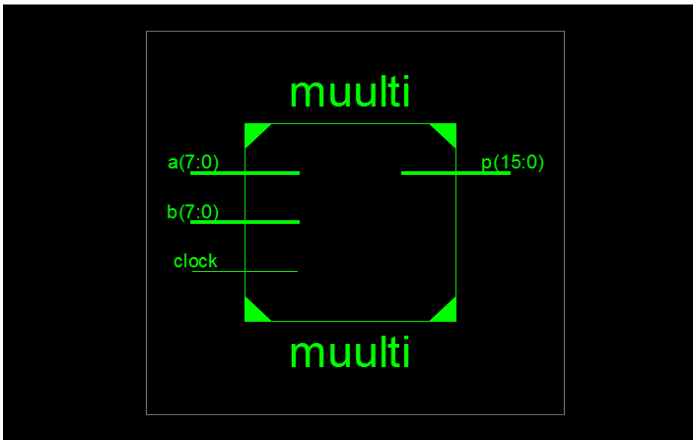

Fig.2 RTL schematic of Radix-4 Booth Multiplier

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 09, September-2016

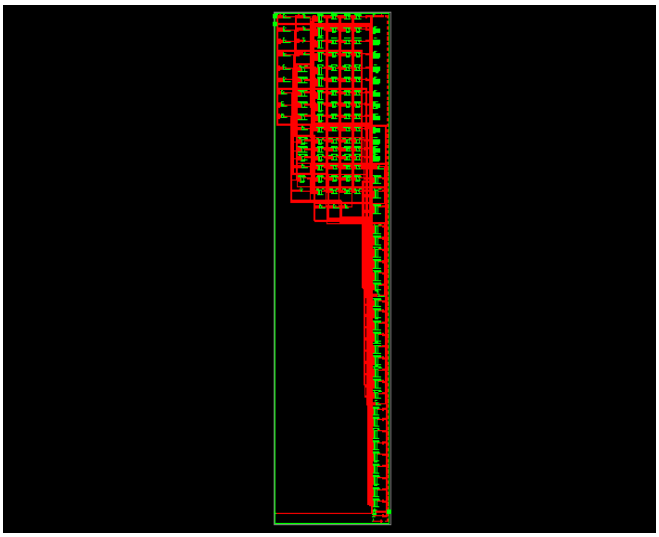

Fig.3 Internal RTL Schematic

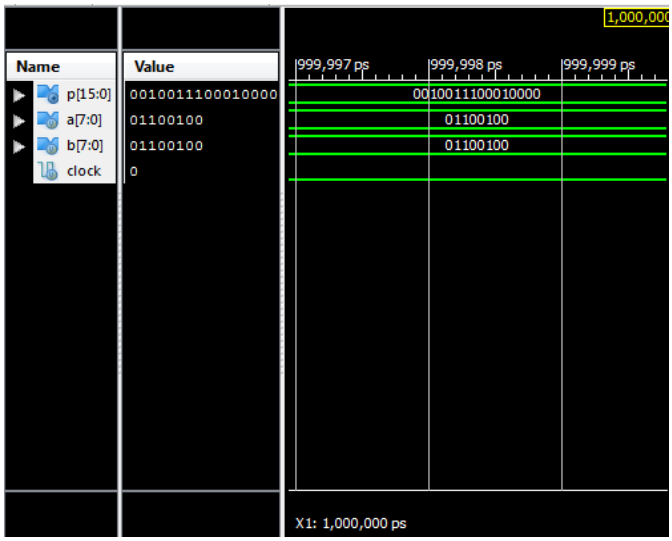

Fig.4 Simulation result of Radix-4 multiplication for unsigned number

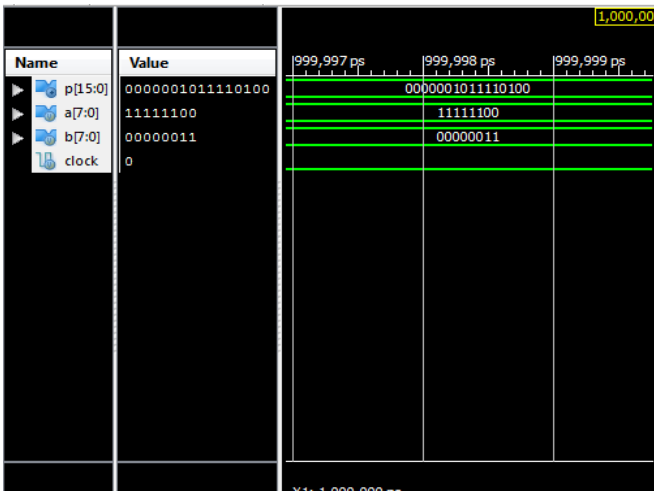

Fig.5 Simulation result of Radix-4 multiplication for signed number

TABLE III. DEVICE UTILIZATION OF RADIX-4 BOOTH MULTIPLIER

Logic Utilization	Used	Available	Utilization
Number of 4 input LUTs	169	7,168	2%
Number of occupied slices	86	3,584	2%
Number of slices containing only related logic	86	86	100%
Number of bonded IOBs	33	141	23%

The multiplication based on Radix-4 Booth algorithm has been simulated on ISim simulator of Xilinx 13.4 software and implemented on FPGA platform for which above results are obtained. Table III gives device utilization information of Radix-4 Booth multiplication.

V. CONCLUSION

When taken into consideration the examples of Radix-2 and Radix-4 multiplication, it can be concluded that, Radix-4 Booth multiplication halves the number of partial products and helps to increase the speed of multiplication operation. This algorithm can be extended to Radix-8 for which complexity is somewhat high, but the generated partial products will reduce to ‘n/3’.

ACKNOWLEDGMENT

I am sincerely thankful to my project guide Prof. Anil G. Sawant who has always been guiding and motivating throughout the project time. I could not have achieved desired objective without his support. It has been a great pleasure for me to work under his guidance.

I am also proud to thank Prof. V. S. Hendre**,** Head of our Department, for approving our project work with great interest.

This project could never have been completed without referring works of many other people whose details are mentioned in references section. I thank everyone and acknowledge my indebtedness to all these people.

REFERENCES

[1] Sukhmeet Kaur, Suman and Manpreet Signh Manna, “Implementation of Modified Booth Algorithm (Radix 4) and its Comparison with Booth Algorithm (Radix 2)”, Advance in Electronic and Electric Engineering, Vol. 3, No.6, pp.683-690, 2013.

[2] Shubhi Shrivastva, Pankaj Gulhane, “Optimized model of Radix-4 Booth Multiplier in VHDL”, International Journal of Emerging Technology and Advanced Engineering, Vol.4, Issue 9, September 2014.

[3] Prof .V .R. Raut, P. R .Loya, “FPGA Implementation of Low Power Booth Multiplier Using Radix-4 Algorithm”, International Journal of Advanced Research in Electrical, Electronics and Instrumentation Engineering, vol.3, Issue 8, August 2014.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 5 Issue 09, September-2016

[4] K. Babulu, G. Parasuram, "FPGA Realization of Radix-4 Booth Multiplication Algorithm for High Speed Arithmetic Logics", International Journal of Computer Science and Information Technologies, vol.2 (5), 2011.
[5] Bodasingi Vijay Bhaskar, Valiveti Ravi Tejesvi, Reddi Surya Prakash Rao, "Implementation of Radix-4 Multiplier with a parallel MAC unit using MBE Algorithm", International Journal of Advanced Research in Computer Engineering and Technology, vol.1, Issue 5,July2012.
[6] Wai-Leong Pang, Kah-Yoong Chan, Sew-Kin Wong, Choon-Siang Tan, "VHDL Modeling of Booth Radix-4 Floating Point Multiplier for VLSI Designer's Library" WSEAS TRANS. on SYSTEMS. Issue 12, Vol. 12, December 2013.
[7] Rashmi Ranjan, Pramodini Mohanty, "A New VLSI Architecture of Parallel Multiplier Based on Radix-4 Modified Booth Algorithm using VHDL", International Journal of Computer Science and Engineering Technology (IJCSET), vol.3, No.4, April 2012.
[8] Khalid Javeed, Xiaojun Wang, Mike Scott, "Serial and Parallel Interleaved Modular Multipliers on FPGA Platform", 2015 25th International conference on Field Programmable Logic and Applications (FPL), pp.1-4.
[9] S.Shafiulla Basha, Syed. Jahangir Badashah, "Design and Implementation of Radix-4 Based High Speed Multiplier using Minimal Partial Products", International Journal of Advances in Engineering & Technology, vol.4, Issue 1, pp.314-325, July 2012.
[10] A. B. Pawar, "Radix-2 Vs Radix-4 High Speed Multiplier", International Journal of Advanced Research in Computer Science and Software Engineering, Vol.5, Issue 3, March 2015.