Redundant Radix-4 Arithmetic Coprocessor Design Using VHDL

Ashish Manoharrao Ingale, Ameed Mustafa Shah

Abstract: With the growth of VLSI processing in the industrial sector the design of efficient algorithms for designing compact functional circuits has led to a competition among various industries. Multiplication is basically a shift add operation. There are, however, many variations on how to do it. Some are more suitable for FPGA use than others. In the area of designing fast parallel algorithms for multiplying numbers, proposed algorithm for multiplying two *n*-bit signed binary numbers needs é 2.71 log2 nù + 3 steps of single bit addition on an *n ´n* systolic architecture which outperforms the then best VLSI implementable algorithm with *O(n)* time and $O(n^2)$ hardware. The subsequent algorithms proposed by him for multiplying numbers in ternary and redundant-radix-four (RR-4) representations require still less time with *2* élog*2n* ù + *2* and é*(1/2)* log*2n +1*ù steps of single digit addition, respectively. Here we have proposed a novel approach for the multiplication of two numbers in RR4 number system. The results has been evaluated in ISE environment and the performance giving satisfactory results.

Keywords: VLSI, RR4, FPGA, MULTIPLIER, COPROCESSOR

1. Introduction

A binary multiplier is an electronic circuit used in digital electronics, such as a computer, to multiply two binary numbers. It is built using binary adders. A variety of computer arithmetic techniques can be used to implement a digital multiplier. Most techniques involve computing a set of partial products, and then summing the partial products together. This process is similar to the method taught to primary schoolchildren for conducting long multiplication on base-10 integers, but has been modified here for application to a base-2 (binary) numeral system. In binary encoding each long number is multiplied by one digit (either 0 or 1), and that is much easier than in decimal, as the product by 0 or 1 is just 0 or the same number. Therefore, the multiplication of two binary numbers comes down to calculating partial products (which are 0 or the first number) shifting them left, and then adding them together (a binary addition, of course):

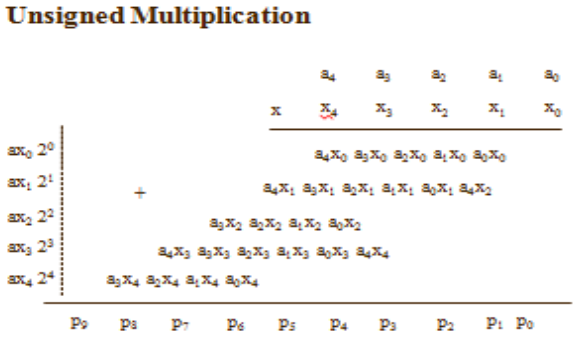


Fig 1.1: *Unsigned Multiplication*

- *Ashish Manoharrao Ingale Research Scholar, M.Tech, Electronic system and communication, Government college of Engineering, Amravati. Maharashtra, India.*
- *Ameed Mustafa Shah Assistant Professor of Department of Electronics and Telecommunication, Government college of Engineering, Amravati. Maharashtra, India.*

1.1 General Requirement

The requirement is to design an 4-by-4 bit multiplier based on the shift and add method. The multiplier shall accept as inputs an 4-bitmultiplier and 4-bit multiplicand as well as a Start signal. The multiplier shall then calculate the result using the shift and add method and provide the 8-bit result along with a Stop signal. The design shall be coded in verilog and simulated for proper functionality and timing.

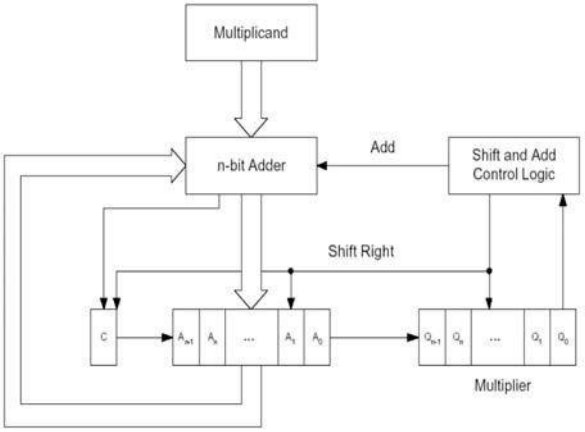


Fig 1.2: *Conventional Multiplication Approach*

2. Problem Identification

In recent times, designing coprocessors for parallel fast multiplications of numbers has become an important field of research. Several algorithms have been developed for multiplication of binary numbers that are easily being implemented on a VLSI chip. For example, Nakamura in proposed an algorithm for iterative array multiplication that requires O(n) time to multiply 2 n-bit binary numbers and can be implemented on a VLSI chip using an almost regular interconnection structure among the processing element. The main problem during the design is the power consumption issues, the LUTs od the design, memory Usage, elapsed time etc .In our design we are implementing a new design showing better results in every aspects.

3. Proposed Technique

3.1 Multiplication in Redundant Radix-4 Number System

The multiplication technique proposed by De and Sinha [1] using radix-4 number representation uses one of the signed- digit (SD) number representation introduced by Avizienis[10] to multiply two m-digit numbers in RR-4 system. We will first discuss about the RR-4 number system. Next the algorithm for An example explaining the steps involved in the partial product generation

$(-3)_{RR-4} = 111$
$(-2)_{RR-4} = 110$
$(-1)_{RR-4} = 101$
$(0)_{RR-4} = 000$
$(1)_{RR-4} = 001$
$(2)_{RR-4} = 010$
$(3)_{RR-4} = 011$

The representation of any RR-4 number using binary bits can be visualized by a matrix of 0's and 1's as follows, where each column represents the respective RR-4 digit:

$$(2\ -1\ 0\ 3\ 1)_{RR-4} = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{pmatrix}$$

The topmost bit in each column indicates the sign of the digit, where 0 stands for positive and 1 stands for negative digit.

Multiplication in RR4 Number System

Multiplicand	1 0 -3 0	First step of partial product	First step of partial
Multiplier	2 -1 1 -2	generation	product generation

-2 0 6 0 { -2 0 2 0 0 -2 1 2 0 ; 0 0 1 0

1 0 -3 0 { 1 0 -3 0 0 1 0 -3 0 ; 0 0 0 0

-1 0 3 0 { -1 0 3 0 0 -1 0 3 0 ; 0 0 0 0

2 0 -6 0 { 2 0 -2 0 0 2 -1 -2 0 ; 0 0 -1 0

Conversion from binary to RR-4 system is described, followed by multiplication algorithm as proposed.

3.2 The RR-4 Number System

In RR-4 number system the radix used is 4 and individual digits belong to the set, S={-3,-2,-1,0,1,2,3}, An m-digit redundant radix

- 4 integer Y=[ym-1….y0] RR-4 ,where for all i, y_i □□{-3,-2,-1,0,1,2,3} and has the value □□$y_i \cdot 4^i$ where i ranges from 0 to (m-1). There are more than one possible representation of the same integer in RR-4 number system. For example, $[0\ 3\ 1]_{RR-4}$, $[1\ -1\ 1]_{RR-4}$, and $[1\ 0\ -3]_{RR-4}$, all represent the number $(13)_{10}$. This redundancy in number representation will be exploited to perform carry propagation
- free addition, thereby allowing the parallel addition of four RR-4 numbers in *O(1)* time, independent of the length of the numbers.

Of the different possible representation of RR-4 digits, one possible way of writing the digits of set S using three binary bits for each digit are as follows, where the leftmost bit is 0(1) if the digit is positive(negative):

1	1	-1	3		Intermediate Sum
1	-1	-1	0		Intermediate Carry
1	0	0	-1	3	Final Sum

4. Results and Discussion

4.1 Device Properties

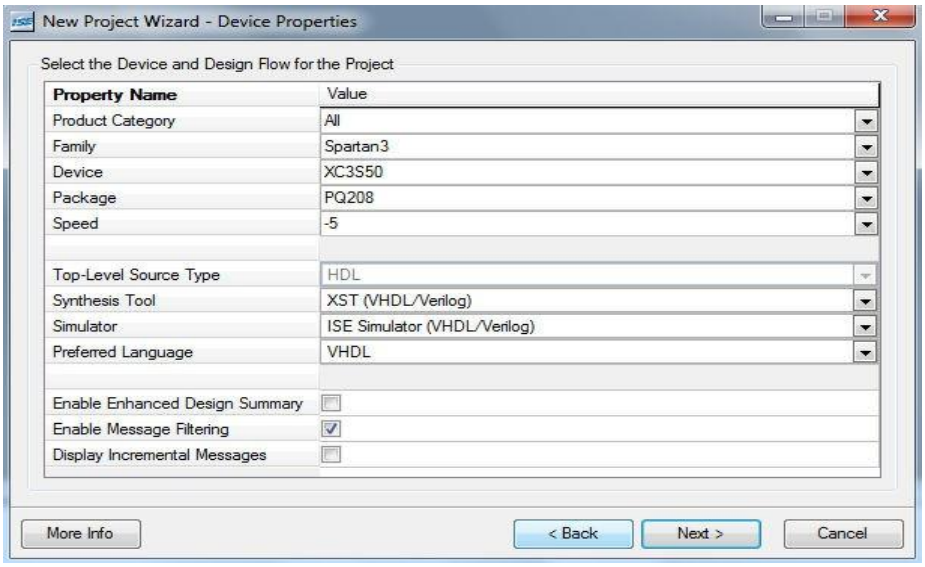


Fig 1.3: *Selected Device properties*

The above figure shows the selected device properties for the synthesis and Simulation in Xilinx ISE environment. We have considered Spartan 3 family with Xc3S50 Device and XC

4.2 HDL Synthesis Report

Macro Statistics

# ROMs	4
32x6-bit ROM	4
# Adders/ Subtractors	16
4-bit adder	16
# Latches	36
1-bit latch	36
# Comparators	8
4-bit comparator greater	8
# Xors	4
1-bit xor2	4

CPU: 12.35 / 13.03 s | Elapsed: 12.00 / 13.00 s
Total memory usage is 172656 kilobytes

4.3 Device Utilization Summary of RR4 Multiplier

Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	75	768	9%
Number of Slice Flip Flops	36	1536	2%
Number of 4 input LUTs	133	1536	8%
Number of bonded IOBs	34	124	27%
Number of GCLKs	1	8	12%

Fig 1.4: *Device Utilization Summaries*

4.4 Power Summary of RR4 Multiplier

4.5 RTL block of RR-4 Multiplier

Power summary:	I(mA)	P(mW)
Total estimated power consumption:		24
Vccint 1.20V:	5	6
Vccaux 2.50V:	7	18
Vcco25 2.50V:	0	0
Clocks:	0	0
Inputs:	0	0
Logic:	0	0
Outputs:		
Vcco25	0	0
Signals:	0	0
Quiescent Vccint 1.20V:	5	6
Quiescent Vccaux 2.50V:	7	18

Thermal summary:

Estimated junction temperature:	26C

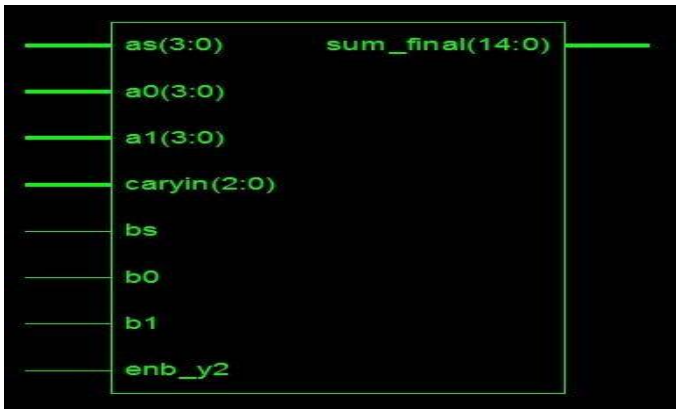


Fig 1.5: *RTL Block for the RR4 Multiplier*

4.6 RTL Schematic of RR-4 multiplier

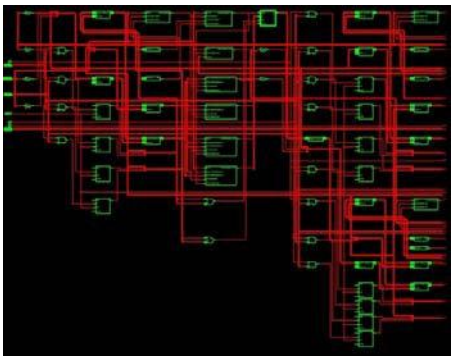
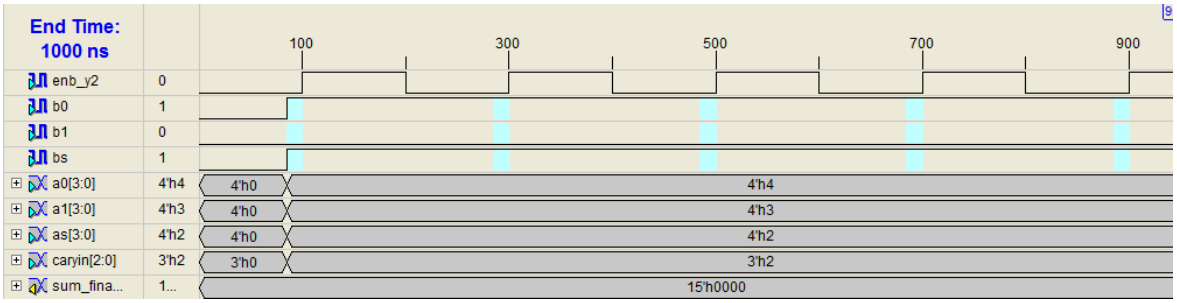
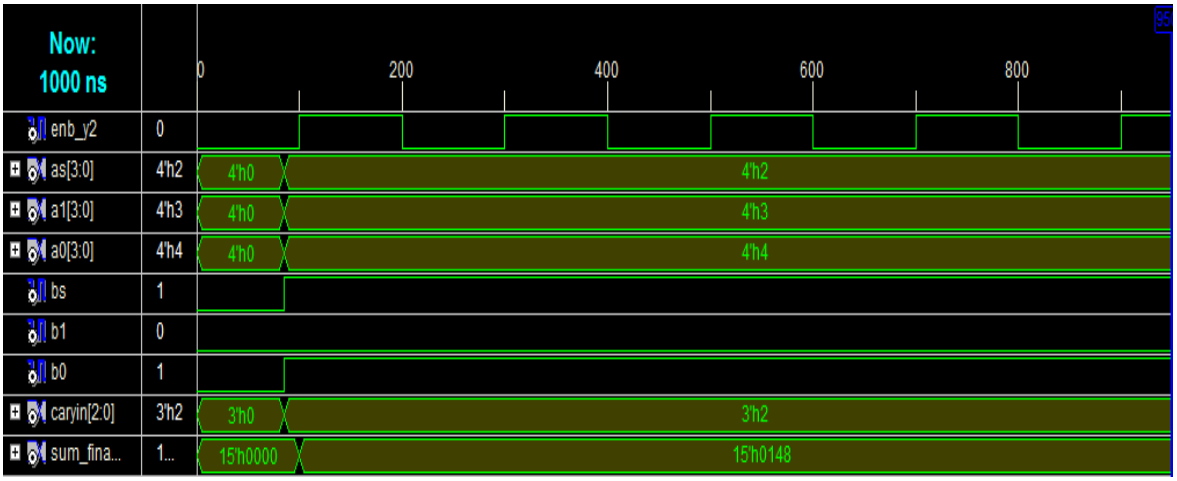


Fig 1.6: *RTL schematic diagram for the RR4 Multiplier*

The above figure shows the synthesized output for the RR4- Multiplier. For Implementing the program in silicon wafer we have to analyze the above synthesis circuit with the no. of components used for the design.(as shown in HDL Synthesis Report).

4.7 Simulations





Above Figure show the simulation for the RR4 multiplier in the Xilinx ISE.WE have multiplied the number in RR4 number Systems. The multiplicand taken here is

As= 0 010
A1= 0110
A0= 0100

=(0 3 -2 0) RR4 Number System

Similarly Bs=1 B1=0 ans B=1 is choosen

Finally we are getting a result of 15'h 148, i.e, 000 0001 01001000 .Which matches with our desired output.

4.8Comparison among different techniques

Parameters	Conventional Multiplier	Redundant Binary Radix-4 Multiplier	Proposed Redundant Radix-4 Multiplier
Power	459	226	124
No. of gates	1416	66	36

We have compare our proposed technique with the existing conventional multiplier as well as with the Redundant Binary Radix-4 Multiplier. From our results we found our proposed scheme has more efficiency with respect to the Power consumption and No .of Gates

5. Conclusions

In this project, we studied the design and methods for RR4 based number System. We proposed a technique towards development of an efficient multiplication of two RR4 numbers. The Process has been simulated using Xilinx ISE Environment and various results like HDL Syenthsis, Device utilization report has been shown along with the simulation results. The technique has a greater advantages for various VLSI designs and it can be extended and used in design an efficient coprocessor in future works.

6. References

[1] Sriharish, L., & Kamaraju, M. (2014). A Novel VLSI Architecture of Multiplier on Radix 4 using Redundant Binary Technique. International Journal of Computer Applications, 103(2), 23-28.

[2] Antelo, E., Villalba, J., Bruguera, J. D., & Zapata, E. L. (1997). High performance rotation architectures based on the radix-4 CORDIC algorithm. Computers, IEEE Transactions on, 46(8), 855-870.

[3] Li, C. C., & Chen, S. G. (1997, April). A radix-4 redundant CORDIC algorithm with fast on-line variable scale factor compensation. In Acoustics, Speech, and Signal Processing, 1997. ICASSP-97., 1997 IEEE International Conference on (Vol. 1, pp. 639-642). IEEE.

[4] Seo, Y. H., & Kim, D. W. (2010). A new VLSI architecture of parallel multiplier–accumulator based on Radix-2 modified Booth algorithm. Very Large Scale Integration (VLSI) Systems, IEEE Transactions on, 18(2), 201-208.

[5] He, Y., Chang, C. H., Gu, J., & Fahmy, A. H. (2005, May). A novel covalent redundant binary Booth encoder. In Circuits and Systems, 2005. ISCAS 2005. IEEE International Symposium on (pp. 69-72). IEEE.

[6] Wu, H., Hasan, M. A., Blake, I. F., & Gao, S. (2002). Finite field multiplier using redundant representation. Computers, IEEE Transactions on, 51(11), 1306-1316.

[7] M. De and B. P. Sinha, "Fast Parallel multiplication using redundant quarternary number system", Parallel Processing Letters, Vol. 7, pp. 13-23 1997.

[8] Alodeep Sanyal, Rajat Shuvra Ghoshal, Achintya Das and Susmita Sur-Kolay A Reconfigurable Coprocessor for Redundant Radix-4 Arithmetic.

[9] M. De and B. P. Sinha, "Fast Parallel multiplication using redundant quarternary number system", Parallel Processing Letters, Vol. 7, pp. 13-23 1997.

[10] S. Nakamura, "Algorithms for iterative array multiplication", IEEE Trans. Comput., Vol.35, pp.713- 719, 1986.

[11] N. Takagi, H. Yassura, S Yajima, "High Speed VLSI multiplication algorithm with a redundnt binary addition tree" IEEE Trans. Comput., Vol. 34, pp. 789-796, 1985.

[12] B. P. Sinha and P. K. Srimani, "Fast parallel algorithms for binary multiplication and their implementation on systolic architectures", IEEE Trans. Comput., Vol. 38, pp. 424- 431, 1989.

[13] M. De and B. P. Sinha, "Fast parallel algorithm for ternary multiplication using multivalued I2 L technology", IEEE Trans. Comput., Vol. 43, pp. 603-607, 1994. R. P. Brent and H.T. Kung, "A regular layout for parallel adders", IEEE Trans.

[14] Comput., Vol. 31, pp. 260-264,1982.

[15] K. Mehlhorn and F. P. Preparata, "Area-time optimal VLSI integer multiplier with minimum computation time", Information and Control, Vol. 58, pp. 137-156, 1983.

[16] Karatsuba and Y. Ofman, "Multiplication of multi-digit numbers on automata," Soviet Physics Doclady, Vol. 7, pp-595-596, 1963.

[17] N. Takagi and S. Yajima, "Modular multiplication hardware algorithms with a redundant representation and their application to RSA cryptosystem," IEEE Trans. Comput., Vol. 41, pp. 887-891, 1992.

[18] Avizienis, "Signed-digit number representation for fast parallel arithmetic," IRE Trans. Electron. Comput., Vol. EC-10, pp. 389-400, 1961.

[19] P. J. Ashenden, The designer's guide to VHDL. San Francisco, California: Morgan Kaufman Publishers Inc. 1996.

[20] V. Vetz, J. Rose, A Marquardt, Architecture and CAD for deep-submicron FPGAs. USA: Kluwer Academic Publishers, 1999.

[21] Z. Salcic, VHDL and FPLDs in digital systems design, prototyping and customisation. USA: Kluwer Academic Publishers, 1998.

[22] Xilinx Data Book, 2000.