TEST Engineering & Management

January - February 2020
ISSN: 0193 - 4120 Page No. 11186 - 11190

A FPGA Approach of Coprocessor Designing Using RR4 Algorithm

Dr. Anil Kumar Sahu[1], Abhilasha singrol[2]
[1]SSGI (FET), SSTCBhilai, India,[2] SSGI, SSTC Bhilai, India
anilsahu82@gmail.com, singhrolabhilasha@gmail.com

Article Info
Volume 82
Page Number: 11186 - 11190
Publication Issue:
January-February 2020

Article History
Article Received*: 18 May 2019*
Revised*: 14 July 2019*
Accepted*: 22 December 2019*
Publication*: 21 February 2020*

Abstract

Now a days as VLSI industry is growing fast the design of an efficient algorithm for designing compact functional circuits has led to a competition among various industries. Multiplication is basically a shift operation. However there is various methods for perform this operation. Some are more suitable for FPGA use than others. For implementing fast multiplication of binary numbers parallel schemes will be used. Algorithm for multiplication of two n-bit signed binary number needs e2.71 log2n + 3 steps of bit by bit addition n`n systolic architecture that gives the best result along with the VLSI implementable scheme with 0(n) computational time and 0(n2) hardware requirements. The algorithm proposed for multiplying numbers in ternary and redundant –radix-4 (RR-4) representations require minimum time with 2log 2n +2 and (1/2) log2n+1u steps of single digit addition. Here we demonstrate the addition of numbers without any carry- propagation time causes significant decrease in the multiplication time.While comparing with conventional and other methods it will reduce power consumption up to 36 mW and also reduce the number of gates. In this proposed method the no of gates will be 456, which is minimum as compared to other methods.

Keywords: *RR4, Multiplier, Coprocessor, Fpga, Xilinx, Spartan-3 And Spartan 6 family.*

I. INTRODUCTION

RR4 arithmetic coprocessor is 8 bit RICS processor and it is having two different unit. 1) Arithmetic logic unit (ALU) 2) Bidirectional interface unit. The RR4 ALU consist of binary to RR4conversion unit, arithmetic unit, logic unit, control unit and top level module. In Bidirectional interface unit set of routines in C++ that communicates between PC and XS40 board in both direction. VHDL design module that control the interfacing between FPGA and PC. The parallel port need to be in SPP mode due to control of each control and status line from the routine of the software, otherwise nStrobe line will toggle during the port is written and n program signal will erase the current programming targe[1]t. In digital electronics a binary – multiplier is an important circuit which is used in computer, for the multiplication operations of two or more digital numbers. This mainly comprises of binary adders. A lot of computer arithmetic scheme may be utilized for a digital multiplier. Most of the schemes includes analyzing and evaluating a set of partial products, again adding the partial products together [2-3]. This method is similar to the technique explained to the primary student for doing multiplication on base of 10, however it has been changed a base-2 (binary) number system for various applications. In binary number system each digit is multiplied along with number (either 0 or 1) and it is easier with comparison to decimal process, again product by 0 or 1 is just a string of 0 or 1. Hence the multiplication process of two binary numbers is adding the 0 and 1 them together to get result with required shift operation. The key of arithmetic operations in such applications is multiplication and the development of fastest multiplier circuit has been a subject of interest over decades[5-8]. Reducing the power consumption and delay time are

January - February 2020
ISSN: 0193 - 4120 Page No. 11186 - 11190

essential for much application. Multiplier based on RR4 mathematics is one of the fast and low power multiplier[9-11].

II.BASIC PROVISIONS

Based on the shift and add method the basic requirement is to design a 4x4 multiplier. With the start signal, multiplier shall accept 4-bit as an input of multiplier and 4-bit multiplicand. By using shift and add method multiplier should calculate the result and provide 8-bit result along with stop signal. For proper functionality and time design shall be coded in VHDL.

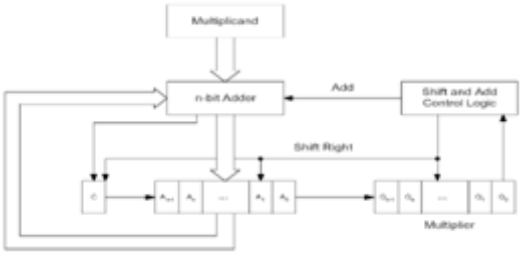


Figure 1.: Shift and Add Multiplication

The IC designers and hardware designers reuse their implementation because of increase in complication of algorithm in every area of electronics and integrated circuit. Company and designer cannot meet the expense of develop any design from scratch, which increases the demand of cheap and sophisticated products. It shows the importance of "macro" level for point that reusability and specifically designs [6].

In today market high demand of lower prices, fastest speed of High speed multipliers, filter structure with precise function created to produce a flexible product.In arithmetic operation, multiplication is an important fundamental function. Multiplication-based operation such as multiply and accumulation (MAC) and inner product are among some of frequently used Computation –Intensive Arithmetic Function (CIAF) currently implemented in many Digital Signal Processing (DSP) application such as convolution, Fast Fourier Transform (FFT) , filtering in microprocessors in its arithmetic and logic . There is no need of high speed multiplier

A.PROBLEM STATEMENT

Now a day, designing a coprocessor for fast multiplication of numbers is a main concern. There are different algorithm present in which binary numbers can be easily multiplied and implemented on VLSI chip .The main problem in design is power consumption and large number of gates which introduces delay. In our design we are implementing a new design showing improved results in every aspect.

III. PROPOSED WORK

Here we planned for design a coprocessor using redundant radix 4 (RR4) algorithm and evaluating various performance and results. Here we compare all conventional method with the proposed design, all the details are given in the below table. From our results we found that our proposed scheme has more efficiency with respect to the power and number of gates used.

A.ARCHITECTURE OF COPROCESSOR

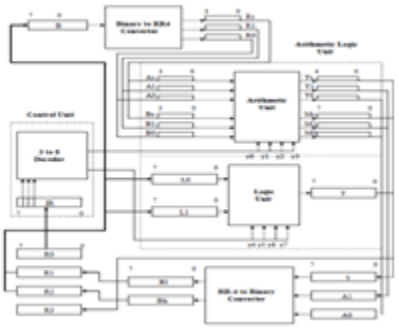


Figure 2.:Architecture of Coprocessor

B. BINARY TO RR4 CONVERSION UNIT:

During the arithmetic operation in RR4 ALU first 8 bit operand is converted into its 12 bit equivalent RR4 representation and after completion of operation it's reconverted into binary bits. For binary logic in VHDL codes described in RR4 bin.vhd and full-adder.vhd. The outputs for addition, logical or multiplication operation are redesigned to the operands registers of the BIU

C.ARITHMETIC UNIT:

The basic arithmetic operations using the RR4 algorithm as discussed earlier.

i) Addition

January - February 2020
ISSN: 0193 - 4120 Page No. 11186 - 11190

ii)Subtraction

iii) Multiplication

iv)Complementation

E. MULTIPLICATION OF RR4 NUMBER SYSTEM:

Multiplication technique proposed by De [1] using radix-4 number demonstration uses a signed number to multiply m-digit numbers in RR4 system shown in Fig. 3. Most complex and time consuming operation of the coprocessor is multiplication and it operates by the repeated execution of the modules RR-4 multiplier.vhd and partial –product.vhd. The output digits will be taken from the partial products. Once the binary number is transform into its RR-4 equivalent representation, the multiplication of two (m x m) number in RR-4 in [(1/2) log2 m] +1 computational steps.

Let A and B is the RR-4number which is multiplied. Then Pi =A, is the ith partial product. The ith partial product is stated as the sum of two vector Ci (1) and Ci (2) fig.4.2. The resultant sum vector is denoted by Di. be reduced.

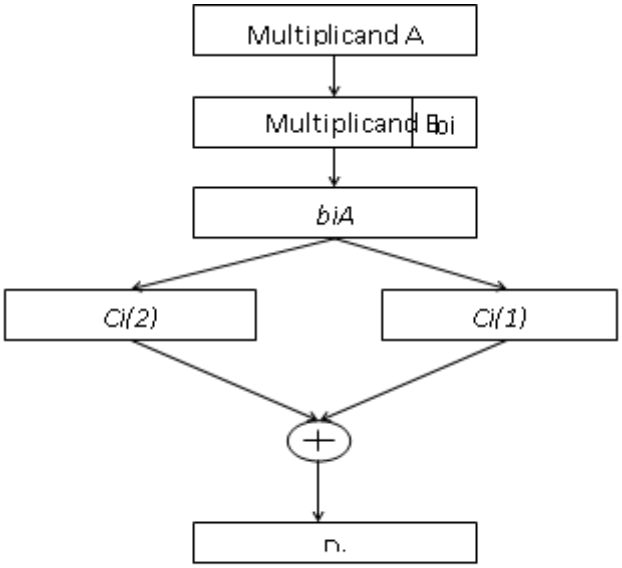


Figure 3.: Generation of Partial Product of (n+1)

F. RR-4 Number System

Radix 4 is used in RR-4 number system and digit belong to set, S= {-3,-2,-1, 0, 1, 2, 3}, and m-digit RR-4 integer Y= {ym-1…..y0} RR-4, where for all i, yi {-3, -2 -1, 0, 1, 2, 3} and has the value yi.4i where i ranges from 0 to(m-1). RR-4 representation of number is not unique. Which means any number can be represented in more than one ways. For example, [1-1-2], [030] both represent (12)10 this redundancy in representation of numbers introduce extra addition thereby parallel addition is done in which there is no effect on length of the numbers.

G. Logic Unit

The logic unit performs basic logic operations:

i) AND

ii)OR

iii)NOT

iv) X-OR

The logic circuit are nothing to do with the RR4 representation of the number. The logic unit takes two 8-bit operands performs bitwise logic operations and produces an 8-bit output.

H. Control Unit:

The control unit of the coprocessor is simple and consist of an instruction register (IR) and 3-to-8decoder unit. The decoder generate 8 control signal.

I. Top Level Module

In this module coprocessor encapsulates all the separate units describes in it and provides the interconnection between the separate units.

J. Bidirectional Interface Unit

1. A set of routines in C++ that communicates between the XS40 board and the PC in both directions.

2. VHDL design modules that control interface between PC and FPGA.

January - February 2020
ISSN: 0193 - 4120 Page No. 11186 - 11190

K. INSTRUCTION SET OF THE COPROCESSOR

The RISC coprocessor accomplish 8 different operation whose command words are given below. Which is shown in Fig. 4.

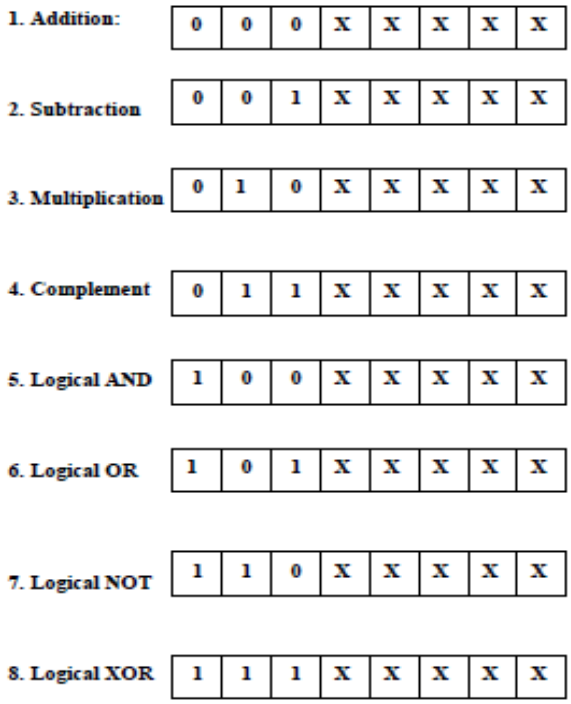


Figure 4: Risc Coprocessor

VI. RESULTS AND DISCUSSION

As we have presented a figure 5 shows the selected device properties for the synthesis and simulation in Xilinx ISE en-vironment. We have consider Spartan 3 family with Xc3S100s Device and XC. Here table I shows the device utilization summary and Table II also compared the results with jageshwarresult. With the help of comparison table we can easily consider the improvement of the newly proposed method. This figure also includes the number of slices, LUTs, flip-flop, IOBs, Globle clock. The result may be further used for comparison with other system with same parameter.

TableI. Device Utilization Summary (estimated values)

Parameters	Conventional Multiplier	Redundant Binary Radix-4 Multiplier	Proposed Redundant Radix-4 Multiplier
Power (mW)	45.9	44	36
No. of gates	1416	1200	456

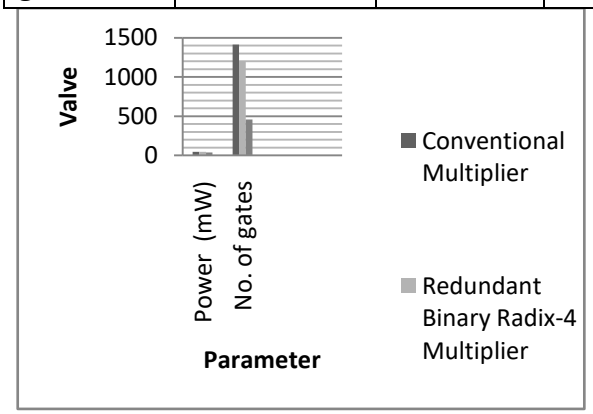


Figure 5.: comparison of methods in terms of power consumption and no of gates used.

A. HDL Synthesis report of the coprocessor

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

==========

B. RTL Schematic Arithmetic Unit

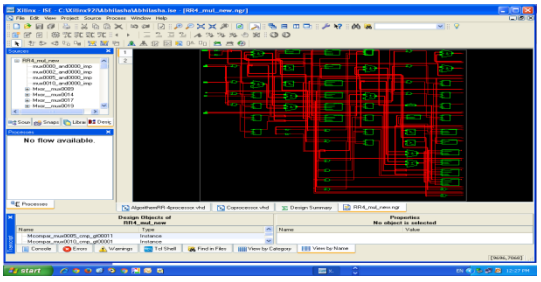
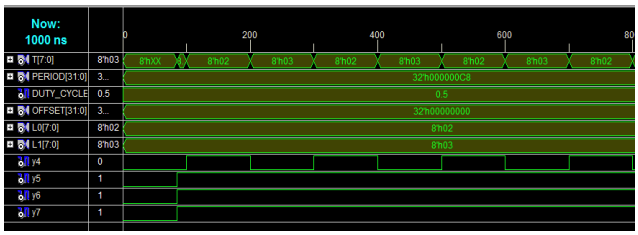


Figure 6.:RTL Schematic Arithmetic Unit

January - February 2020
ISSN: 0193 - 4120 Page No. 11186 - 11190

C.Simulation for theLogic Unit of Coprocessor



D.Simulations Results for Arithmetic Unit

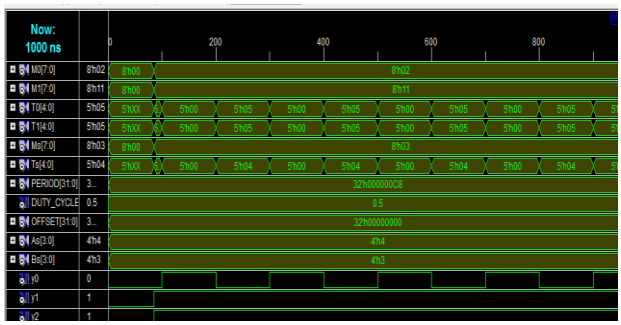


Figure 8.: Simulations Results for Arithmetic Unit

Table II.

Comparative Results

Logic utilization	Used in proposed method	Available	Utilization of proposed method	Used in jageshwar method	Available	Utilization of jageshwar method
Number of slices	75	960	7%	1206	2352	51%
Number of 4 input LUTs	133	1920	1%	417	4704	8%
Number of slice flip-flop	36	1920	6%	2171	4704	46%
Number of 4-input LUTs	133	1920	6%	2171	4704	8%
Number of bonded IOBs	34	66	51%	56	142	39%
Number of GCLKs	1	24	4%	4	4	100%

V.CONCLUSION

In the proposed work, we have first designed a RR4multiplierand compared the results in terms of no. of gates and power consumption, finally compared the results with the conventional multiplier and redundant binary radix4 multiplier, we found our proposed RR4 based scheme has greater efficiency with all aspects of result. Subsequently we have extended our work in designing 8 bit arithmetic coprocessor with the help of redundant radix-4 arithmetic number system. This co-processor is supposed to work on a simple parallel quaternary multiplication algorithm which is the fastest known in the domain of parallel computing. The co-processor is able to perform arithmetic operations like addition, subtraction, multiplication and complementation, and logical operations like logical AND, logical OR, logical NOT and logical XOR .We have successfully synthesized the results and done the simulations in the Xilinx ISE.

REFERENCES

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

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

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

[4] Shigeo Kuninobu ; Tamotsu Nishiyama ; Hisakazu-Edamatsu ; Takashi Taniguchi ; Naofumi Takagi, “Design Of High Speed MOS Multiplier And Divider Using Redundant Binary Representation”,(1987),IEEE 8th Symposium on Computer Arithmetic (ARITH) 10.1109/ARITH.1987.6158706

[5] Yajuan He, Chip-Hong Chang, JiangminGu “A Novel Covalent Redundant Binary Booth Encoder”, (2005), IEEE.0-7803-8834-8/05/.

[6] Seo, Y. H et al. “A new VLSI architecture of parallel multiplier–accumulator based on Radix-2 modified Booth algorithm”. VLSI Systems, 18(2), IEEE Transactions on 201-208.

[7] AshishManoharrao et al. “redundant radix-4 arithmetic coprocessor design using VHDL”, (2016), ISSN2277-8616.

[8] ShindeAbhijeet S, JambhaleVidyadhar S, DeoreGaurav P, Sivanantham S and Sivasankaran. K, “A Reconfigurable Coprocessor Units with Redundant Radix -4 Arithmetic” (2015), ICGET 987-1-4673-8625-8.

[9] Alodeepsanyal , Rajatshuvraghoshal, Achintya das “A Reconfigurable Coprocessor for Redundant Radix -4 Arithmetic”(2011)

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

[11] S. Nakamura, “Algorithms for iterative array multiplication”, 1986, IEEE Trans. Compute., Vol.35, pp.713-719.