International Journal of Reconfigurable and Embedded Systems (IJRES)
Vol. 1, No. 1, March 2012, pp. 37~42
ISSN: 2089-4864

Simplified VHDL Coding of Modified Non-Restoring Square Root Calculator

Tole Sutikno[1], Aiman Zakwan Jidin[2], Auzani Jidin[3] and Nik Rumzi Nik Idris[4]
[1]Departement of Electrical Engineering, Universitas Ahmad Dahlan, Yogyakarta, Indonesia
[2] IP Design Department, Altera Corporation (M) Sdn Bhd, Penang, Malaysia
[3]Department of Energy Converstion, Universiti Teknologi Malaysia, Johor Bahru, Malaysia.
[4]Department of Power Electronics and Drives, Universiti Teknikal Malaysia Melaka, Melaka, Malaysia.

Article Info

Article history:

Received Jan 14, 2012
Revised Mar 10, 2012
Accepted Mar 26, 2012

Keyword:

FPGA
Non-Restoring Algorithm
Pipelined Architecture
Square Root Calculation

ABSTRACT

Square root calculation is one of the most useful and vital operations in digital signal processing, the operation which in recent generations of processors is performed by the hardware. The hardware implementation of the square root operation can be achieved by different means, but it is very dependent on programmer's sense and ability to write efficient hardware designs. This paper offers universal and shortest VHDL coding of modified non-restoring square root calculator. The main principle of the method is similar with conventional non-restoring algorithm, but it only uses subtract operation and append 01, while add operation and append 11 is not used. The strategy has been conducted to implement it successfully in FPGA hardware, and offer an efficient in hardware resource, and it is superior.

Corresponding Authors:

Tole Sutikno,
Departement of Electrical Engineering,
Universitas Ahmad Dahlan,
Kampus III UAD, Jln. Prof. Soepomo, Janturan, Yogyakarta 55164, Indonesia.
Email: tole@ee.uad.ac.id

1. INTRODUCTION

In many VLSI applications, it is an urgent requirement to provide the computation of square root of a binary coded number with low power dissipation and fast computation (low delay propagation). Square root calculation is one of the most useful and vital operations in computer graphics and scientific calculation applications, such as digital signal processing (DSP) algorithms, math coprocessor, data processing and control, and even multimedia applications [1-6]. It is a classical problem in computational number theory, which is oftenly encountered and which is a hard task to get an exact result [7-8].

Many square root calculation techniques have been proposed, such as Rough estimation, Babylonian method, exponential identity, Taylor-series expansion algorithm, Newton-Raphson method, Sweeney Robertson Tocher redundant and non redundant method, restoring and non-restoring algorithm (digit-by-digit method) [1-9]. However, the early processors carry out the square root operation of the algorithms above by software means, which have long delays for its completion [6]. With the rapid advancement of technology which allows the integration of large circuits on a single chip and the increase in demand for faster computational execution time, the hardware realization of square root became more attractive [6]. Unfortunately because of the complexity of the square root algorithms, the square root calculation is not easy to be implemented on Field Programmable Gate Array (FPGA) technology [1, 3, 5, 10].

There are some algorithms of the square root computation which are already implemented on FPGA. They are generally grouped into two distinct categories. The first category is called estimation methods, which includes algorithms such as Rough estimation and Newton-Raphson method (and also its derivations:

CORDIC, DeLugish's and Chen's), whereby the second category is called digit-by-digit method. The restoring algorithm has a big limitation at restoring step in the regular flow. Primarily for this reason, although initially having led the way for all the other methods, it has been declined in importance and nowadays it is no longer used [11]. The non restoring algorithm does not restore the remainder, which can be implemented with least hardware resource usage. It is the most suitable for FPGA implementation and allows for IEEE standard rounding to be readily implemented [1-3, 6].

Many strategies or architectures have conducted to implement the non restoring digit-by-digit square root algorithm in FPGA hardware. Yamin and Wanming [1-2, 9] have introduced a non restoring algorithm with fully pipelined and iterative version that requires neither multipliers nor multiplexors. They introduced the carry save adder (CSA) and the carry propagate adder (CPA) as basic building blocks. Although the algorithms in [1-2] have a good processing speed, they consume too many hardware resources as a trade-off, while the algorithms in [9] has low computation speed, despite costing less resource usage. The similar architectures have been introduced by Xiaoliang [10], Thakkar [12] and Xiumin et al [13]. In the other study, Samawi et al [6] have introduced controlled add-sub (CAS) as basic building blocks. The effort is done to reduce hardware consumed, with moderate delay. The other architecture which has also been proposed is a fully combinational architecture [4]. However, FPGA is very suitable to adopt fully pipelined architecture because of the characteristics of its structure. Hence, , very little or even needless extra cost is required, if the pipeline technology is implemented in FPGA [14].

In this paper, a strategy to implement non restoring square root algorithm based on FPGA which adopt fully pipelined architecture, will be presented. The main principle of the method is only uses subtract operation and append 01 which is implemented in register transfer level (RTL) abstraction, but add operation and append 11 are not used. In the proposed strategy will needs fewer pipeline stages compared with the proposed algorithm in [12]. Next, the performance of the developed design will be compared to the one developed by Samawi et al [6].

2. MODIFIED NON-RESTORING SQUARE ROOT ALGORITHM

Samavi, et al [6] has improved classical non-restoring digit-by-digit square root circuit by eliminating redundant blocks which still based on constant binary digit of 01 or 11 and adder-subtractor as the main building block. This paper offers a simple strategy while only uses subtract operation and appends 01. This strategy is implemented by VHDL programming at RTL abstraction.

A hardware implementation of the non-restoring digit-by-digit algorithm for 6-bit unsigned square root by an array structure is shown in Figure 1. The radicand is P (P5,P4,P3,P2,P1,P0), U (U2,U1,U0) as quotient and R (R4,R3,R2,R1,R0) as remainder. It can be shown that the implementation needs three-stage pipelines. The basic building blocks of the array are blocks called Controlled Subtract-Multiplex (CSM). Figure 2 presents the details of a CSM. The inputs of the building block are x,y,b and u, while ports bo(borrow) and d (result) are the outputs. If u=0, then d<=x-y-b; else d<=x. For optimizing hardware resource utilization of the implementation above, specialized entities can be created as building block components. It will eliminate circuitry that is not needed.

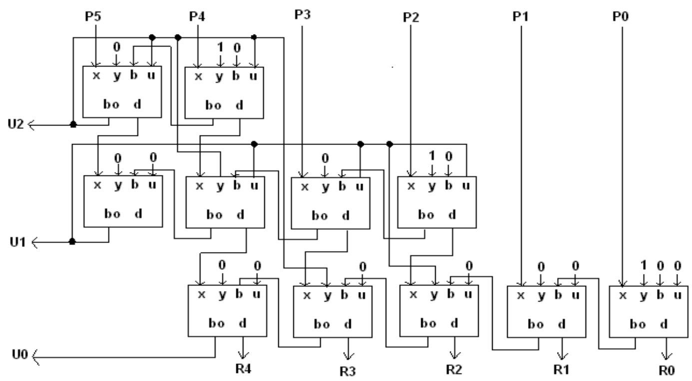


Figure 1. A simple hardware implementation of the non-restoring digit-by-digit algorithm for unsigned 6-bit square root

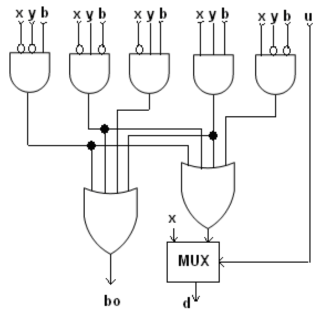


Figure 2. Internal structure of a CSM block

In platform of register transfer level (RTL) abstraction, the above can be described as follow:

Step 0. Start

Step 1. Initialization of the radicand (the n-bit number will be squared root), the quotient (the result of squared root), and the remainder. To calculate the square root of a 2n-bit number, it needs n stage pipelines to implement the proposed algorithm.

Step 2. At the binary point, divide the radicand into groups of two digits in both direction (integer and fractional??).

Step 3. Beginning on the left (the most significant bit), select the first group of one or two digit (If n is odd then the first groups is one digit, and vice versa) (the 2n-bit radicand is always even??)

Step 4. Choose 1 squared, and then subtract.
First developed root is “1” if the result of subtract is positive, else “0”

Step 5. Shift two bits, subtract guess squared with append 01.
N^{th}-bit squared is “1” if the result of subtract is positive, and because of subtract operation is done
else
N^{th}-bit squared is “0”, and not subtraction has been performed

Step 6. Repeat Step 5 until end group of two digits

Step 7. End

3. PROPOSED VHDL CODING

This paper proposes a universal and shortest VHDL coding of modified non-restoring square root calculator as shown below. In fact, this RTL code is easy-to-use and parameterizable, since the input radicand size can be modified by only setting the appropriate n value.

```
library ieee;
use ieee.numeric_std.all;
use ieee.std_logic_1164.all;

entity m_sqrt is
generic ( n: positive:= 32 );    -- n: number of bits for output; 2n: number of bits for input
  {port declaration}
end entity;

architecture RTL of m_sqrt is
  {variable declaration}
  ...
begin
    remain(0) := (others=>'0');    --r0 = 0
    qint  (0) := (others=>'0');    --q0 = 0
    r(0) :=  (others=>'0');
    for i in 1 to n loop
    if  (signed(remain(i-1)) >= 0) then
        r(i) := remain(i-1)(n-1 downto 0) & (input(2*(n-i+1)-1 downto 2*(n-i+1)-2));
    else
        r(i) := r(i-1)(n-1 downto 0) & (input(2*(n-i+1)-1 downto 2*(n-i+1)-2));
    end if;
    q(i):= qint(i-1)(n-2 downto 0) & "01";
    remain(i) := std_logic_vector ( unsigned(r(i)) - unsigned(q(i)));
    qint(i)   := qint(i-1)(n-1 downto 0) & not(remain(i)(n+1));
    end loop;
    output <= qint(n)(n-1 downto 0);
end process;
end RTL;
```

Figure 3. Universal and shortest VHDL coding of modified non-restoring square root calculator

Simulation and hardware experiments have been conducted to validate the VHDL code. The code is implemented and evaluated based on Altera DE2 FPGA, as shown in Figure 4. To observe the output calculation of the square root, they are connected to an 8-bit ADC system.

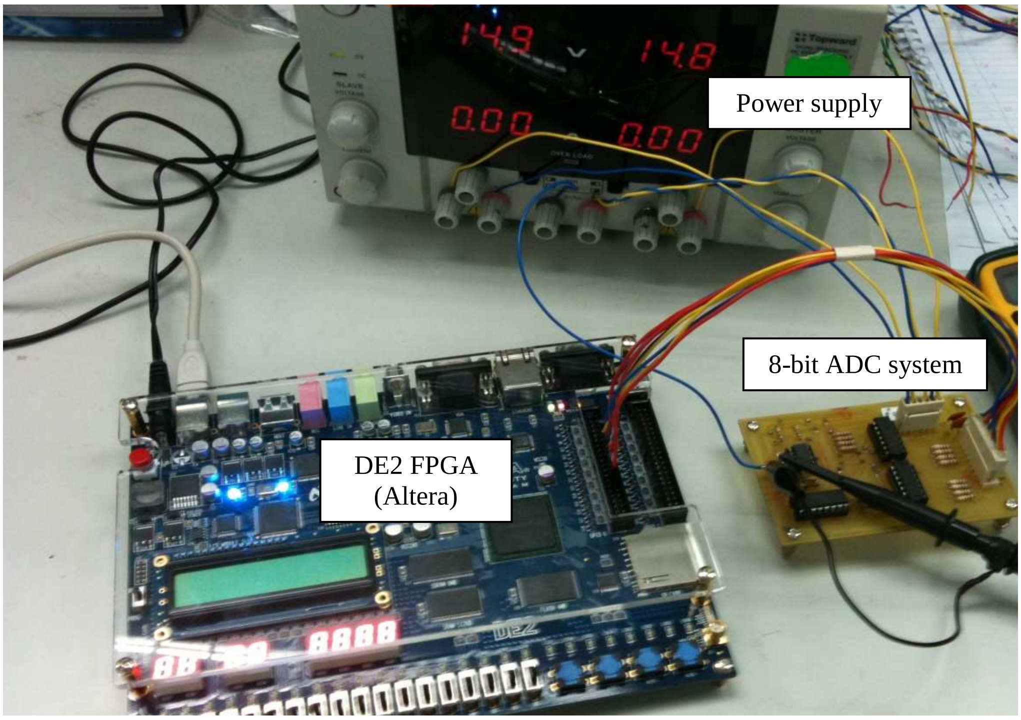
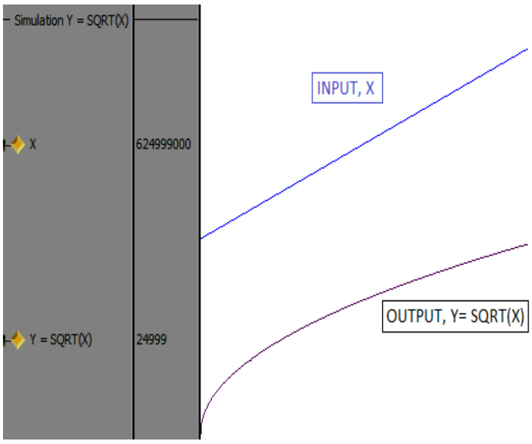


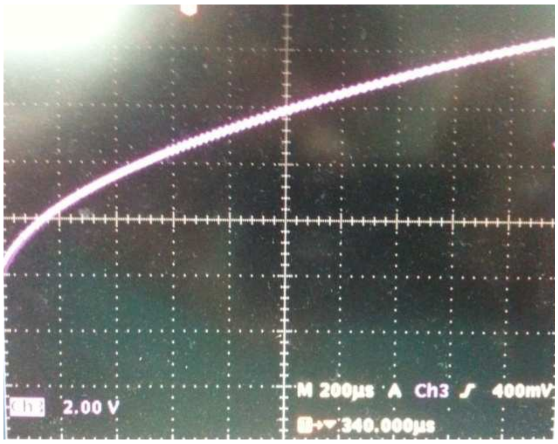
Figure 4. Experiment setup

4. RESULTS AND ANALYSIS

In the previous sections, the hardware implementation of the non-restoring digit-by-digit algorithm for square root is described. The first observation is conducted to validate the output of the square root calculation in simulation, which has been performed by using ModelSim-Altera. Then, the results acheived are reproduced in the hardware test, by observing the ADC system output. The simulation and experiment result are shown in Figure 5.a and 5.b, respectively.



a. simulation result



b. hardware result

Figure 5. Simulation and hardware implementation results

The more detailed simulation result is shown in Figure 6. For example, the calculation of square root of 257 is 16.0303. The computation error is very small (< 10^{-3}), and the error level is acceptable. The results showed that the implementation has succeeded and worked properly.

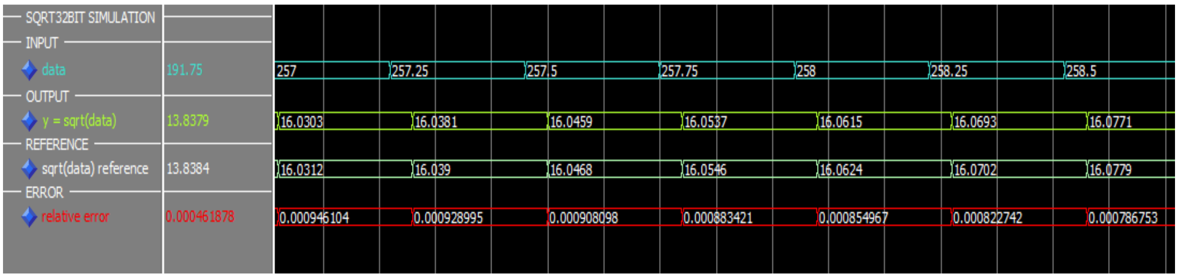


Figure 6. The more detailed simulation result

The performance of the proposed VHDL code is shown in Table 1. The numbers of the logic elements (LE) used are obtained from Quartus II compilation report. It has shown a fantastic value for reducing of hardware resource consumed compared to references [6] and [16]. This is due adoption fully pipelined architecture and also simplification of the VHDL code.

Table 1. Performance of the proposed VHDL code

Bits	Input wordsize	Input range	Output	Output range	Precision	LE used
8	[4.4]	0.00 - 15.9375	2.2	0.00 - 3.75	0.25	16
16	[8.8]	0.00 - 255.996	4.4	0.00 - 15.9375	0.0625	99
32	[16.16]	0.00 - 65535.99998	8.8	0.00 - 255.996	0.0039	360
32	[10.22]	0.00 - 1024	5.11	0.00 - 31.9995	4.9e-4	360
56	[28.28]	0.00 - 2^28	14.14	0.00 - 16383.99994	6.1e-5	1072
64	[32.32]	0.00 - 2^32	16.16	0.00 - 65535.99998	1.5e-5	1395

5. CONCLUSION

In many VLSI applications, it is an urgent requirement to provide the computation of square root. The operation is one of the most useful and vital operations in digital signal processing. This paper has presented a novel strategy of the FPGA implementation of non restoring square root calculator. It has provided a universal and shortest VHDL coding of modified non-restoring square root calculator, and offers an efficient in hardware resource, and it is superior.

ACKNOWLEDGMENT

The authors wish to acknowledge the Research Management Center (RMC) of Universiti Teknologi Malaysia (UTM) for the financial funding and providing instrumentation devices support of this project.

REFERENCES

[1] L. Yamin and C. Wanming, "Implementation of Single Precision Floating Point Square Root on FPGAs," in *IEEE Symposium on FPGA for Custom Computing Machines*, Napa, California, USA, 1997, pp. 226-232.
[2] L. Yamin and C. Wanming, "Parallel-array implementations of a non-restoring square root algorithm," in *Computer Design: VLSI in Computers and Processors, 1997. ICCD '97. Proceedings., 1997 IEEE International Conference on*, 1997, pp. 690-695.
[3] K. Piromsopa, *et al.*, "An FPGA Implementation of a fixed-point square root operation," presented at the Int. Symp. on Communications and Information Technology (ISCIT 2001), ChiangMai, Thailand, 2001.
[4] D. R. Llamocca-Obregon, "A Core Design to Obtain Square Root Based on a Non-Restoring Algorithm," presented at the IBERCHIPS Workhsop, Salvador Bahia, Brazil, 2005.
[5] XiaojunWang, "Variable Precision Floating-Point Divide and Square Root for Efficient FPGA Implementation of Image and Signal Processing Algorithms," Doctor of Philosophy, Electrical and Computer Engineering, Northeastern University, Boston, Massachusetts, 2007.
[6] S. Samavi, *et al.*, "Modular array structure for non-restoring square root circuit," *Journal of Systems Architecture*, vol. 54, pp. 957-966, 2008.
[7] H. Dong-Guk, *et al.*, "Improved Computation of Square Roots in Specific Finite Fields," *Computers, IEEE Transactions on*, vol. 58, pp. 188-196, 2009.
[8] S. Lachowicz and H. J. Pfleiderer, "Fast Evaluation of the Square Root and Other Nonlinear Functions in FPGA," in *Electronic Design, Test and Applications, 2008. DELTA 2008. 4th IEEE International Symposium on*, 2008, pp. 474-477.
[9] W. Chu; and Y. Li;, "Cost/Performance Tradeoff of n-Select Square Root Implementations," in *5th Australasian Computer Architecture Conference (ACAC 2000)*, Canberra, ACT 2000, pp. 9-16.

[10] J. Xiaoliang, "Implementation of Square Root Arithmetic Based on FPGA," *Modern Electronics Technique,* vol. 30, 2007.

[11] P. Montuschi and M. Mezzalama, "Survey of square rooting algorithms," in *Computers and Digital Techniques, IEE Proceedings E,* Italy, 1990, pp. 31 - 40.

[12] A. J. Thakkar and A. Ejnioui, "Design and implementation of double precision floating point division and square root on FPGAs," in *Aerospace Conference, 2006 IEEE*, 2006, p. 7 pp.

[13] W. Xiumin*, et al.*, "A New Algorithm for Designing Square Root Calculators Based on FPGA with Pipeline Technology," in *Hybrid Intelligent Systems, 2009. HIS '09. Ninth International Conference on*, 2009, pp. 99-102.

[14] G. Renxi*, et al.*, "Hardware Implementation of a High Speed Floating Point Multiplier Based on FPGA," in *4th International Conference on Computer Science & Education*, Nanning, Guangxi, P.R.China, 2009.

[15] S. Dattalo. (2000, March 17, 2010). *Square Root Theory*. Available: http://www.dattalo.com/technical/theory/sqrt.html

[16] March 30, 2010). *Comparing Altera APEX 20KE & Xilinx Virtex-E Logic Densities*. Available: http://www.altera.com/products/devices/apex/features/apx-compdensity.html

BIOGRAPHY OF AUTHORS

Tole Sutikno received his B.Eng. and M.Eng. degree in Electrical Engineering from Diponegoro University, Indonesia and Gadjah Mada University, Indonesia, in 1999 and 2004, respectively. Since 2001 he has been a lecturer in Electrical Engineering Department, Universitas Ahmad Dahlan (UAD), Indonesia. Currently he is pursuing PhD degree at the Universiti Teknologi Malaysia (UTM), Malaysia. His research interests include the field of digital design, power electronics, motor drive systems and FPGA applications.

Aiman Zakwan Jidin received his B.Eng, and M.Eng. degree from Ecole Supérieur d'Ingénieur en Electronique et Electrotechnique Paris (ESIEE Paris). His research interests include Field Programmable Gate Array (FPGA) applications and digital electronic design. In July-August 2008 and May–August 2010 he is accepted for his Assistant Engineer Internship at Universiti Teknologi Malaysia (UTM) Skudai, Johor, Malaysia and in Jan–June 2011 he worked as an Engineer Intern at Logic Design Solutions, France. Currently, he is a Design Engineer at Altera Corporation (M) Sdn Bhd, Penang, Malaysia

Dr. Auzani Jidin received his B.Eng., M.Eng. and PhD degree in Power Electronics & Drives from Universiti Teknologi Malaysia (UTM), Malaysia in 2002, 2004, and 2011 respectively. He is a lecturer in Department of Power Electronics and Drives, Faculty of Electrical Engineering at Universiti Teknikal Melaka Malaysia (UTeM), Malaysia. His research interests include the field of power electronics, motor drive systems, FPGA and DSP applications.

Dr. Nik Rumzi Nik Idris received the B.Eng. degree in Electrical Engineering from the University of Wollongong, Australia, the M.Sc. degree in power electronics from Bradford University, West Yorkshire, U.K., and the Ph.D. degree from Universiti Teknologi Malaysia (UTM) in 1989, 1993, and 2000, respectively. He is an Associate Professor at the Universiti Teknologi Malaysia, and an Administrative Committee Member of the Industry Applications Societies/Power Electronics/Industrial Electronics Joint Chapter of IEEE Malaysia Section. His research interests include ac drive.