Novel Square Root Algorithm and its FPGA Implementation

G.Anupama, A.Raghuram

***Abstract*— Square root is a basic arithmetic operation which is used in digital signal processing. Due to complex algorithm it is difficult to implement on FPGA. This paper presents a novel square root algorithm which is based on some ancient Indian mathematics (Vedic mathematics) formula called Dwandwa Yoga. The proposed algorithm uses lowest area. In this paper we are using 24-bit (16bit+8bit) floating point input and 16-bit (8bit+8bit) floating point output. We will discuss the proposed square root algorithm, its hardware description, and its FPGA implementation using Xilinx tool. The cost for this algorithm on the SPARTAN-3E XC3S500E is 173 LUTs (4 input), 90 slices, 99 slice flip flops. The design consumes 90.9 mW power and can be operated at frequeney 68.22MHz. This algorithm uses only adders, subtractors and registers whieh leads to consume less area, less power and high operating frequencies. The proposed algorithm can be implemented on deeimal numbers.**

***Index Terms*— Dwandwa Yoga, FPGA, PARTAN-3E, Square root, Vedic Mathematics, Xilinx.**

I. INTRODUCTION

Arithmetic operations like addition, subtraction,ultiplication and division; square root is also an important arithmetic operation in digital signal processing. The squareroot unit is also an important part of 3D graphics, scientific calculations, communication systems, audio processing units, image processing units, spectrum analyzers and many others.
Square root unit is very difficult to implement in digital hardware because of its complexity. Other problems with the implementation of square root unit are time taken to calculate square root, operating frequency, area and power consumption. So a new algorithm and hardware circuit is required to meetabove constraints.

Many algorithms have so far been proposed to calculate square root of a number. These are NR method, SRT redundant method, non-redundancy method. Also there are some other architecture like parallel array architecture, pipelined architecture and low cost architecture, CORDIC algorithm.
A lot of research is carried out to implement the square root algorithms on FPGA. The non-restoring algorithm for pipelined architecture is implemented on FPGA. The square root algorithm is implemented and using CORDIC algorithm. And many more algorithms are also implemented on FPGA.

G.Anupama, Dept of Electronics&Communication Engineering Indira institute of technology&Sciences,Markapur
A.Raghuram,Dept of Electronics&Communication Engineering Indira institute of technology&Sciences,Markapur

In this paper we will discuss our proposed algorithm based on Vedic mathematics. Then we will design hardware circuitry for 24 bit floating point input and \6 bit floating point output using the proposed algorithm. Finally we will discuss the simulation results and required hardware, power consumption, operating frequency etc.

II. PREVIOUS WORK

In previous researches, there are many methods and algorithms for computation of square roots; some of them are discussed below:

A. Newton-Raphson method:

Newton-Raphson is an iterative method [4] to calculate the square root of a number. The original formula for square root is given below(1).

QK+1=1/2(QK+D/QK)-------(1)

Q_{k+1}= the output in (k+ 1)th iteration, Q_k= the output in Kth iteration(or square root of input D), D= the input data on which square root operation is to be done.

Some derivative formula of NR method is also used to calculate $Y=\sqrt{X}$, this time we use the iteration equation given below in (2):

$Ti+1=1/2\times Ti\times(3-T_i^2\times X)$-----(2)

Ti= approximate value of $1/\sqrt{X}$ after ith iteration. After n iteration let Tn be the approximate value of $1/\sqrt{x}$, then square root is obtained by equation $Y=\sqrt{X}= T_n\times X$.

B. Restoring Algorithm

Restoring algorithm is a popular method to implementsquare root algorithm. This algorithm calculates square rootand remainder value in iterative process.

$$D = Q^2+R$$

D= input data, Q=square root result and R=remainder. In restoring algorithm we will assume the value of Q and R and calculate the value of Q and R. The algorithm will guess the operand Q for subtraction or addition process to the current R. If our guess was wrong, then we will restore the previous value [5][6].

C Non-Restoring Algorithm

Another method for calculation of square root is nonrestoring algorithm. This algorithm is almost similar to restoring algorithm and used extensively. In this method we will calculate remainder and quotient iteratively.

$$D = Q^2+R$$

D= input data, Q=square root result and R=remainder. In restoring algorithm we will assume the value of Q and R and calculate the value of Q and R. The algorithm will guess the operand Q for subtraction or addition process to the current R. If our guess was wrong, then we will restore the previous value.

III. PROPOSED ALGORITHM

Let the square root of any number X is equal to a three digit number ABC So we have to calculate the value of A, Band C As we can see in the calculation below:
$\sqrt{X}$ = ABC = 100A + l0B + C
X = (100A + l0B + C)2
= $10^4A^2 + 10^3(2AB) + 10^2(B^2 + 2AC) + 10(2BC) + C^2$.

This algorithm is based on ancient Indian mathematics called Vedic mathematics. An operation called as Dwandwa Yoga (DY) is used in this algorithm, the pseudo code for the calculation of DY is given below:

```
DY(Q, k,j)
1) Set DY=0
2) DY=DY+2($q_k$, $q_l$), k=k-l,j=j+l
3) Repeat step 2 until(k>j)
4) If k=j, DY=DY+$q_j$ .
```

Above Pseudo code takes Q, k, j as arguments, where Q =qkqk-1.,qk-2." ..... q_{j+1},q_j, k, j are starting and last bit respectively.

For example: DY of 101 is $(10)_2$ and of 1 1 1 1 is (100)2, We will use this Dwandwa Yoga in our square root algorithm.Now using this Dwandwa Yoga module we will propose square root algorithm. So the square root algorithm is given below:

```
1) Set i=I5,j=7.
2) If $d_i$,$d_{i-1}$ =00 then $q_j$=0 and go to step 3 otherwise go to 4.
3) i=i-2,j=j-1.
4) $q_j$=1, m=j-1, R=di,$d_{i-1}$-0.1.
5) S={R,d $_{i-2}$ }, T=s-10,j=j-1.
6) If T<0, then $q_j$=O, S=T+ 10 else $q_j$=1.
7) R=S,i=i-1 calculate DY(Q, j, m) (Dwandwa Yoga)S=R$d_{i-1}$-D Y
9)If S< 0 then go to step 9 otherwise go to step 12.
I0) j=j+1, i=i+1.
11) Repeat step 9 until $q_j$=0.
12) $q_j$=0 calculate DY(Q,j, m), S=Rd $_{i-1}$-DY
13) T=S-10,j=j-1.
14) Go to step 6 until j> =-8.
```

Where D=dl5dJ./ ..... d,do.d.,d.1 ..... d.7d-8=d,5 $x2^{15}$+ $d_{14}x2^{14}$+.......... +d.7 X)"7+d^{-8} $x2^{-8}$, is 24 bit floating point input. Q =q_7q6 .. q ,qo.q1., ... q.$_7$q-8=q7 $x2^7$+q6 $x2^6$+ .... +q.7X"7+q- 8 $x2^{-8}$.
.DY is Dwandwa Yoga function or operation on (Q, j, m). S, T, R are the temporary registers which are used to store intermediate data. This algorithm is also explained in the flow chart given Fig. 1.

IV. HARDWARE DESCRIPTION

Hardware implementation of this algorithm has reduced the complexity to a large extent. In our proposed algorithm we have used different datapath unit and control unit. The complete circuit for this algorithm is given in Fig. 1. In this circuit we used one 24-bit register for 24 bit floating point input data. In this input, 16 bit are used before decimal point and 8 bit after decimal point. We also used one 16-bit register for 16 bit floating point input data. In this input, 8 bit are used before decimal point and 8 bit after decimal point. These input and output registers are shown by reg Q and reg D respectively in Fig 2. Here we also used one 4 bit counter to point the current position of output register and to decide the termination of the program. It is shown by counter} in this figure 1. Another block of this datapath circuit is DY or Dwandwa Yoga block. In this block the Dwandwa Yoga operation is performed. Now there are some input and output pins for clock (elk), start (start), ready (ready).

The last block of this circuit is the control unit. This is the most important part of this circuit. The control unit is shown by control in Fig. 2. We can understand the working of this control unit using state diagram. The state diagram for control unit is shown in Fig. 3.

In this state diagram total 10 states are there. When start=1 next state will be S I state. In S I state the regD is loaded with the input, the countetj and regQ are cleared in this state. Then the proposed algorithm will compute the square root using the state diagram given in the figure 2.

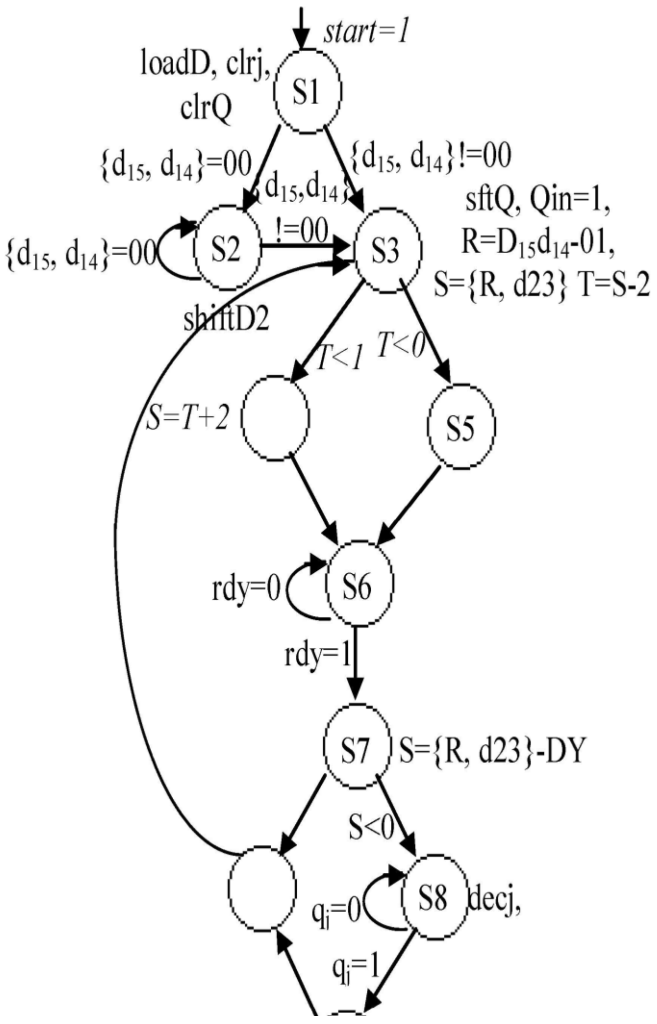

Figure 3. State diagram for control unit

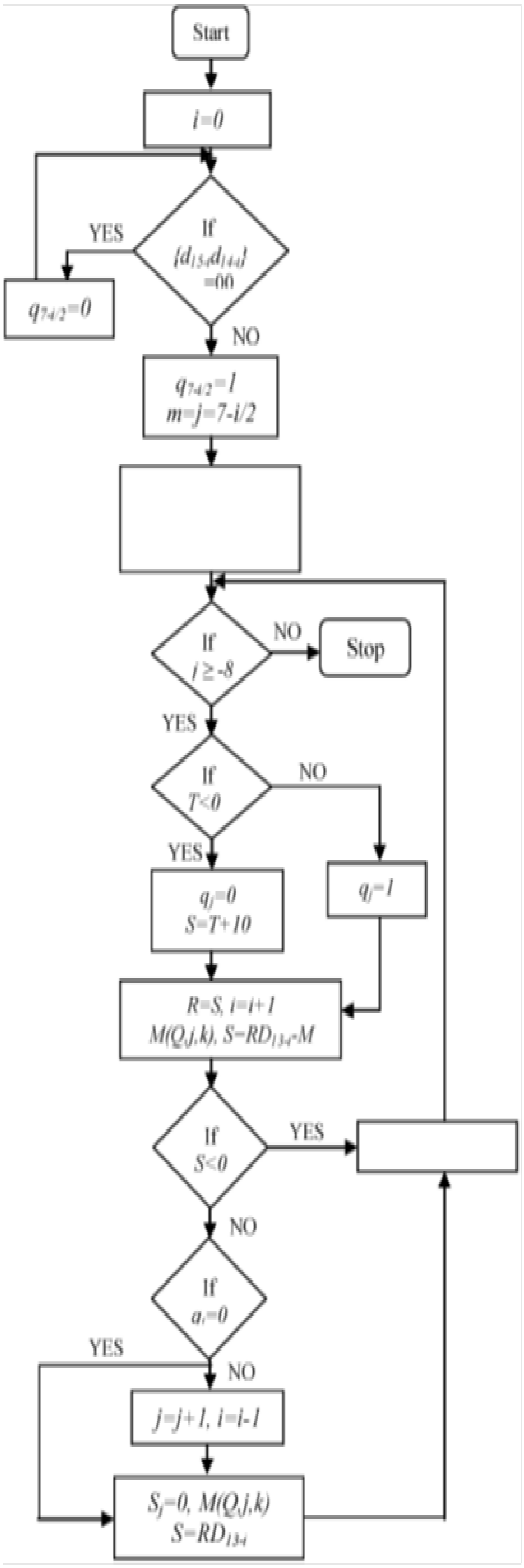

Figure 1. Flowchart for proposed algorithm

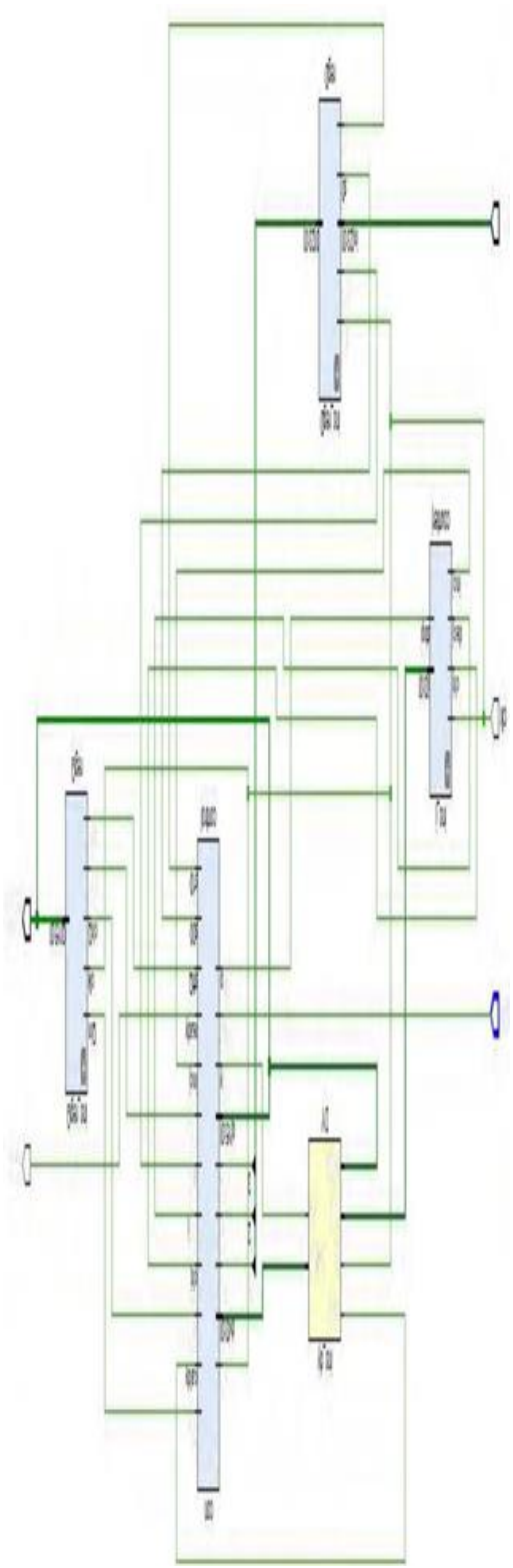

Fig . 2 complete hardware circuit of proposed square root algorithm comprises of datapath circuit and control unit

A. Simulation

Now we will simulate the proposed algorithm and the Dwandwa Yoga block. The results are shown in the Fig. 4 and Fig. 5 respectively. In this figure,
we have given 24 bit input having equivalent decimal value 193.40625. We obtained 16 bit output having equivalent decimal value 13.907. For Dwandwa Yoga, we have given 0111102 as input and obtained the output as 410. We have obtained satisfactory outputs for both blocks.

B.Synthesis:

The proposed designs have been implemented on SPARTAN-3E with FPGA target device XC3S500E-4FG320.In this we have used 8 adders/subtractors, 1 4-bit up-downcounter, 18 registers, 4 latches and 3 multiplexers. The deviceutilization summary is given in Table 1. Timing constraint ofour design is maximum operating frequency is 68.222MHz and minimum period is 14.66 ns. Power analysis of proposed design is tabulated in Table 2 and different power componentsare shown in bar graph in Fig. 7.

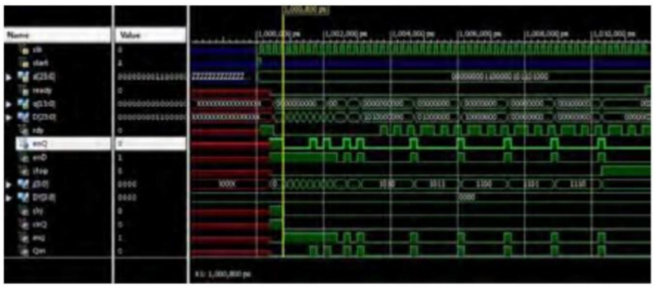

Fig. 4. Output of proposed square root algorithm

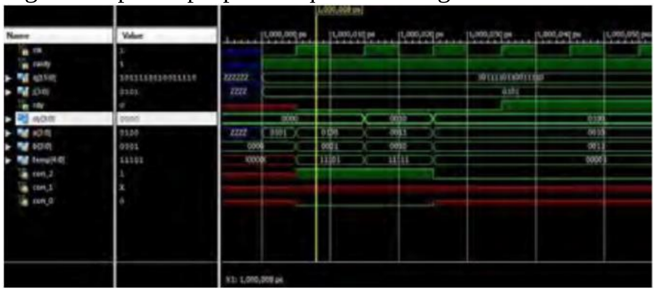

Fig. 5. Output of Dwandwa Yoga (DY) block

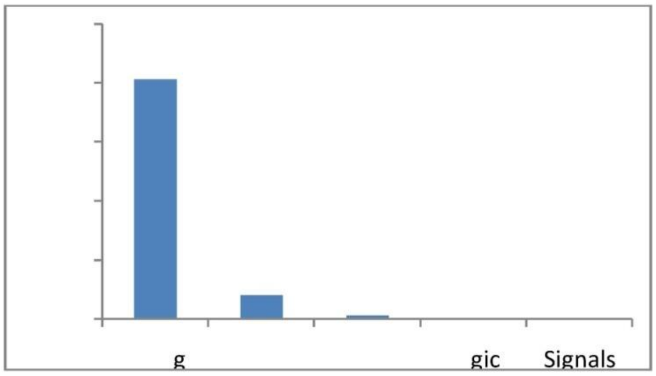

Fig. 6 Different component of power

Table I. Device utilization summary

Component name	No. of component Oft.	%Utilization
Slice	90	l%of4656
Clock frequency	50 MHz	68. 22 MHz (worst case)
Static Power	81 mW	136mW
Dynamic Power	9. 9mW	10mW
Total Power	90. 9mW	146mW

Table 2. Power analysis

Slice flip flops	99	l%of9312
4 input LUTs	173	l%of9312
Bonded JOBs	42	18% of 232
GCLKs	1	4%of 24

VI. CONCLUSION

A new square root algorithm using Vedic Mathematics is presented in this paper and this algorithm is implemented on SPARTAN-3E FPGA. The proposed design is less complexthan the previous designs and produces accurate results for floating point inputs. Hardware design of presented algorithm for 24 bit floating point input requires 90 slices, 99 slice flip flops and 173 LUTs. Timing analysis shows that the maximum operating frequency for this design is 68.22 MHz (14.66 ns).

ACKNOWLEDGMENT

The authors are thankful to Asst.Prof. A. Raghuam.Indira institute of technology and sciences.Markapur for providing environment that we used to obtain the synthesis results. We wish to thank and our well wishers for esteem appreciation.

REFERENCES

[1] www.ripublication.com
[2] www.researchgate.net
[3] Bharati Krsna Tirthaji Maharaja. "Vedic Mathematics" Delhi.India: Motilal Banarsidass, 2012, pp. 295-303.
[4] I. Sajid, M. M. Ahmed, and S. G. Ziavras, "pipelined implementation of fixed point square root in FPGA using modified non-restoring algorithm," in Proc. IEEE 2nd international conf. on Computer and Automation Engineering, 2010, vol. 3, pp. 226-230.
[5] B. Yang, D. Wang, and L. Liu, "Complex division and squareroot using CORDIC," in Proc. IEEE 2nd international conf on consumer electronics, communications and networks, 2012, pp. 2468-2468,2012.
[6] c. Ramamoorthy, J. Goodman, and K. Kim, "Some properties of iterative square rooting methods using high-speed multiplication," IEEE Transactions on Computers, Vol. C-21, No. 8, pp. 837-847, 1972.
[7] Y. Li, W. Chu, "A new non-restoring square root algorithm and its VLSI implementations," Proceedings International Conference on Computer Design : VLSI in Computers and Processors, pp. 538-544, October 1996.
[8] T. Sutikno, "An efficient implementation of the non restoring square root algorithm in gate level," International Journal of Computer Theory and Engineering, vol. 3, pp.46-51, February2011.
[9] Xilinx, Inc., The Xilinx Design Language, April 2013, HTMLdocumentation file supplied with ISE Veri on 14. 5.