SRT FAST DIVISION ALGORITHMS: SIMULATION AND PERFORMANCE EVALUATION

Selvihan Nazlı Yavuzer Ahmet Sertbaş
e-mail: syavuzer@bahcesehir.edu.tr *e-mail: asertbas@istanbul.edu.tr*
Istanbul University, Faculty of Engineering, Department of Electrical & Electronics Engineering, 34850, Avcılar, Istanbul, Turkey

Key words: *Division, , SRT algorithms, high-radix division, performance analysis*

ABSTRACT

SRT division algorithms are common in modern floating-point units. These algorithms process more quotient bits in each step to provide a higher division performance. This study evaluates SRT algorithms (SRT-2, SRT-4, SRT-8 and SRT-16) based on memory and process time requirements. For this purpose, a C# algorithm simulator for SRT algorithms have been implemented and relative algorithm analysis have been performed based on memory and time criteria.

I. INTRODUCTION

Fast division unit design is important in high-speed computing as division –among computer-based arithmetic operations- is the most complex operation with long delay periods. On the other hand, with the IEEE floating point standard that requires exact division, division algorithm design suitable for large micro-processing circuits has been a focus. Various algorithms have been developed based on quadratic and linear approximation algorithms. A significant approximation algorithm is Newton-Raphson [1, 2] method for the first group, while SRT [3] method is common as the linear approximation algorithm.

Most implementations for the division are based on the SRT algorithm that uses a recurrence producing one quotient digit for each step. The speed of such SRT-based dividers is mainly determined by the complexity of the quotient-digit selection. To speed up the division process, one may reduce the number of iteration steps by increasing the radix Q of the process. Selecting $\beta=2^m$ allows the generation of m quotient bits at each step and the number of steps can be reduced to (n/m). However, the complexity of the quotient-digit selection and remainder updating increases for high radices, eliminating the advantages of the reduction in number of iterations [2]. Different methods have been developed to ease the process of division bit selection [4-7]. The simplest method for bit selection is involves the use of a look-up table called quotient-bit selection table. In this method, partial remainder is compared to table values to reach a result. However, as radix increases, the look-up table size also increase which also increases the bit-selection cost.

Since SRT division was proposed, there has been much discussion on the required precision of the truncated partial remainder and divisor for algorithm convergence. There is no general theory on the evaluation of the degree of truncation tolerable in SRT division and so no means of comparing schemes employing different radices [8].

II. SRT DIVISION ALGORTIHMS

The SRT class of division algorithms is characterized by the use of redundant representations for the quotient, and most often as well for the remainder. Since the invention in the late fifties simultaneously by Sweeney, Robertson and Tocher [9] and the introduction of the use of redundant representations for the remainders by D.E. Atkins [1], these methods have been extensively studied and implemented in processors.

Division operation is simply defined as

$$X = q.d + rem \qquad (1)$$

where x is dividend, d is divisor, q quotient, rem (if any) remainder and

$$|rem| < |d| \,.\, ulp \text{ and } sign(rem) = sign(x) \qquad (2)$$

Item length for quotient is determined using ulp (unit in the last place) and following criteria;

- If ulp=1 then quotient is an integer
- If ulp = r – n, where n is the number of quotient bits an r is the radix.

The basic idea behind SRT division is to speed-up the digit selection process using limited number of comparisons based on a few most significant bits of d and w[j]. In SRT iteration method, partial remainder is calculated as shown in (3):

$$w[j+1] \leftarrow r \,.\, w[j] - q_{i+1} \,.\, d \qquad (3)$$

where w[j] represents the partial remainder after j iteration steps starting with w[0] = x. Quotient at jth step is shown in (4)

$$q[j] = \sum_{i=1}^{j} q_i.r - i \qquad (4)$$

In each iteration step, the purpose is to determine the new quotient bit q_{j+1}. In hardware implementations of SRT division, selection is done by scanning a reference table using a few most significant bits of quotient (d) and the partial remainder (w[j]). In the case of a software implementation, tables for quotient digit selection can not be used in order to avoid cache misses [10].

Robertson diagram in Figure 1 shows the remainder vs. quotient bit for SRT division.

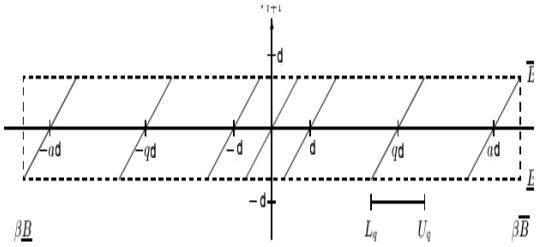

Figure 1. Robertson diagram for SRT division

SRT-2 DIVISION ALGORITHM

SRT-2 division equation for (β=2) is as shown in (5).

$$q_{j+1} = \begin{cases} \bar{1}, if\ 2w[j] < -\frac{1}{2} \\ 0, if\ -\frac{1}{2} \le 2w[j] < \frac{1}{2} \\ 1, if\ 2w[j] \ge \frac{1}{2} \end{cases} \quad (5)$$

At first iteration step where w[0] = x, the quotient should be shifted right by 1 bit providing (2). Figure 2 shows the Robertson diagram for SRT-2 division quotient bit selection.

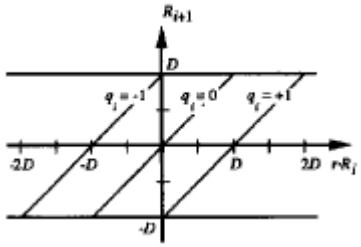

Figure 1. Robertson diagram for Radix-2

SRT-4 DIVISION ALGORITHM

Iteration equation becomes (6) with radix=4.

$$w[j+1] \leftarrow 4 \,.\, w[j] - q_{i+1} \times d \quad (6)$$

And quotient bit is selected among {-2, -1, 0, 1, 2}.

Due to the redundancy in the quotient digit set, there are overlaps between digit selection regions in the Robertson diagram shown in Figure 1, allowing a choice between two digit values. Hence, even if the information on the remainder and divisor is missing, it is possible to select a quotient bit among the alternatives. By allowing such a relaxed quotient digit determination, it is possible to base the quotient digit selection on leading digits of the divisor and of the remainder in a redundant representation. Figure 3 show minimally redundant Robertson diagram for radix-4.

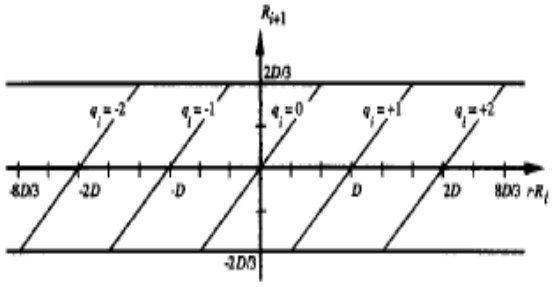

Figure 2. Minimally redundant Robertson diagram for radix-4

SRT-8 DIVISION ALGORITHM

When β=8, recursive SRT equation for remainder at step j+1 is calculated using (7) is used to compute the remainder at j+1, while the quotient bit is selected from {-α, …, -1, 0, 1, …, α} and α in the range, $[(\beta-1)/2] \le \alpha \le (\beta-1)$.

$$w[j+1] \leftarrow 8 \,.\, w[j] - q_{i+1} \times d \quad (7)$$

SRT-16 DIVISION ALGORITHM

To speedup division operations, number of iteration steps can be reduced by using a higher radix. Increasing the radix of SRT division to $r=2^m$ allows the generation of m quotient bits every step [11]. In this manner, the number of required iterations reduces to [n/m], where n is the width of the input operands in bits. However, high-radix division increases the complexity of quotient bit selection and remainder updates, which eliminates the advantage of reduced number of iteration steps. To eliminate this problem, pre-scaling and prediction methods are used. With pre-scaling both dividend and divisor are scaled preserving the quotient. Prediction, involves the selection of new quotient digit is overlapped with the update of the remainder [12].

In SRT-16, quotient bit is selected from the set {-α,…, -1, 0, 1, …, α}, while for α, $15/2 \le \alpha \le 15$ as β = 16. Iteration equation with β =16 becomes (8).

$$w[j+1] \leftarrow 16 \,.\, w[j] - q\,j+1 \times d \quad (8)$$

3. RESULTS

This paper examines the performance changes in division algorithms based radix selection. For this purpose, SRT-2, SRT-4, SRT-8 and SRT-16 division algorithm simulations have been implemented using C3 programming language

(Figure 4). Process time and memory requirements are selected criteria to evaluate performance differences.

Figure 4. SRT Simulator Application

Although SRT-2 algorithm requires less memory space as no loop-up table is needed, the increased number of iteration steps results in a longer execution time when compared to division with 4, 8, 16 radices (Figure 5).

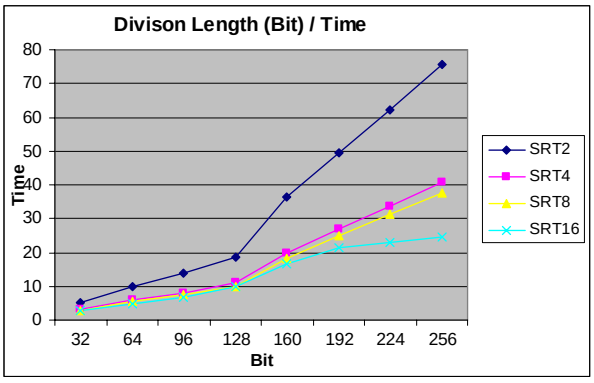

Figure 5. Bit length / time for SRT algorithms

On the other hand, for high radix division algorithms, the size of comparison tables and search cost in reference tables evidently increases as Shown in Figure 6. In Figure 7, process time and memory requirement measures are combined to form an AxT^2 graph which can be used as a broad performance criterion.

Thus, it is possible to use higher radix to obtain a higher-performance algorithm. However, hardware realization costs for this performance gain with higher memory requirements can be assessed through an analysis based on VHDL hardware simulation.

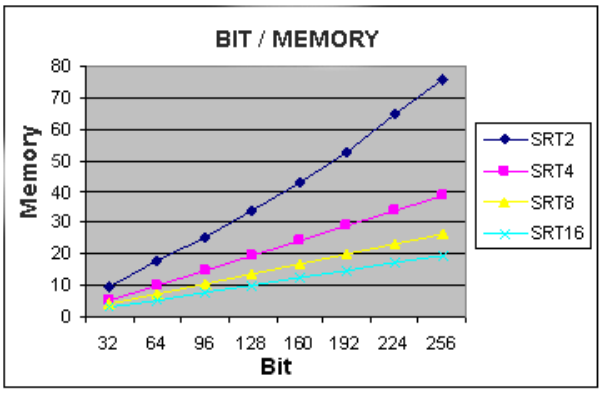

Figure 6. Bit length / memory graph for SRT algorithms

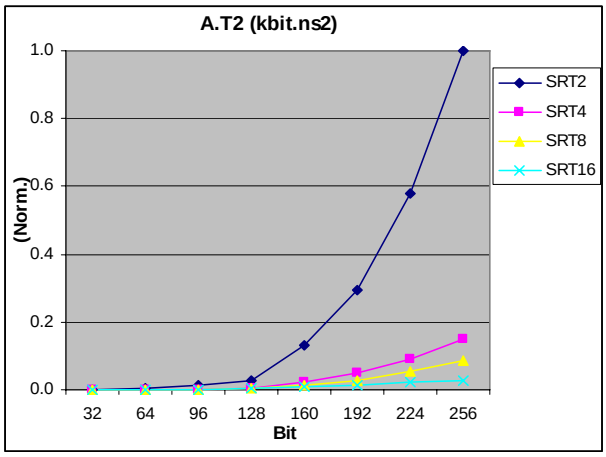

Figure 7. SRT algorithms performance graph (AT^2 - Bit length)

REFERENCES

1. M. D. Ercegovac and T. Lang. Division and Square Root: Digit-Recurrence Algorithms and Implementations. Kluwer Academic Publishers, 1994.
2. I. Koren. , 'Computer arithmetic algorithms' A.K. Peters Ltd., ISBN 1-56881-160-8., 2002.
3. J. E. Robertson., A new class of digital division methods. IRE Transactions on Electronic Computers, EC-7(3):88--92, September 1958.
4. N. Burgess, 'A Fast Division Algorithm for VLSI', Proceedings of the 1991 IEEE International Conference on Computer Design on VLSI in Computer & Processors, p. 560 – 563, ISBN:0-8186-2270-9
5. M.D. Ercegovac, T. Lang, 'Fast Radix-2 Division With Quotient-Digit Prediction', Journal of VLSI Signal Processing, 1, 1989, pp169-180.
6. 'IEEE Standard for Floating Point Arithmetic', IEEE Standard 754, IEEE Computer Society, 1985.
7. K. Hwang, Computer Arithmetic: Principles and Design, J. Wiley and Sons, 1979.

8. T. Williams, N. Burgess, ‘Choices of Operand Truncation in the SRT Division Algorithm’, IEEE Transactions on Computers, Vol. 44 , Issue 7, July 1995, p. 933 – 938, ISSN:0018-9340
9. P. Kernerup, ‘Digit Selection for SRT Division and Square Root’, IEEE Transactions on Computers, Vol. 54, Issue 3, March 2005, p. 294 – 303, 2005, ISSN:0018-9340
10. C. Jeannerod, S. K. Raina and A. Tisserand, ‘High-radix floating-point division algorithms for embedded VLIW integer processors’, 17th IMACS World Congress, Paris, July 2005.
11. T. Pan, H. Kay, Y. Chun, and C. Wey, ‘High-Radix SRT Division with Speculation of Quotient Digits’, Proceedings of the 1995 International Conference on Computer Design: VLSI in Computers and Processors, p. 479, 1995, ISBN:0-8186-7165-3
12. C.-L.Wey and C.-P.Wang, ‘Design of a fast radix-4 SRT divider and its VLSI implementation’, Computers and Digital Techniques, IEE Proceedings, Jul 1999, Vol. 146, Issue. 4, p. 205-210