On Optimal Iterative Schemes for High-Speed Division

E. V. KRISHNAMURTHY

Abstract—This paper describes division schemes which are derived from the classical functional iterative schemes. These schemes are compared with the schemes currently used in the high-speed digital computers.

An error analysis is carried out to reduce the precision requirement in the multiplication operations involved in these division schemes. In addition a scheme is described for selecting the optimal number of minimal precision multipliers for the initial range transformation of the divisor.

Index Terms—Lehman's implementation, measure for efficiency, multipliers, optimal number of multipliers, oscillatory and monotonic convergence, recursions, speed of convergence, table-lookup, truncation effects and precision, Wilkes-Harvard iterative scheme.

Introduction

RECENTLY, there has been a revival of interest in realizing an iterative division scheme as a built-in operation in high-speed digital computers [1]–[4], [5], [10]. Essentially, these iterative schemes belong to the class of schemes described in Wilkes [9] and are familiarly known as Harvard iterative schemes [6]. The reason for implementing this type of iterative scheme is that it can be carried out using only multiplication, complementation, and shift operations. Since a very fast multiplier as well as facilities for simultaneous operations are now available, it is natural that the Wilkes–Harvard scheme turns out to be very convenient for hardware realization.

In this paper we describe alternative division schemes which are basically derived from the classical functional iterative schemes; these schemes have similar features as the Wilkes–Harvard method and turn out to be more general than it. It is shown that these schemes can converge equally fast (or faster) permitting a reduction in the number of entries in the table employed to implement the Wilkes–Harvard scheme [1], [5]. Of course, the price paid for this is in terms of a more complex control requirement.

In order to compare the relative speeds of convergence, a suitable measure is introduced for computing the significance (number of significant digits) of the quotient for a given number of iterations. This is applied to a practical case.

Also, an error analysis is carried out to reduce the precision requirement in the multiplication operations in the preliminary stages of iterations. On the basis of the theoretical results obtained here, a scheme is presented to select an optimal number of minimal precision multipliers required to transform the divisor; this helps the designer to systematically construct the transformation table for a given accuracy and speed.

Manuscript received March 12, 1969; revised August 7, 1969.

The author is with the Department of Applied Mathematics, the Weizmann Institute of Science, Rehovot, Israel, on leave of absence from the Indian Statistical Institute, Calcutta, India.

General Development

The basic principle involved in the iteration scheme is similar to that of Wilkes–Harvard. Let it be required to find $q = a/b$, without remainder. We assume that the operands a and b are p-digit fractions in normalized radix-β form:

$$1/\beta \leq a, b \leq 1.$$

The procedure consists in constructing a sequence of multipliers $m_0, m_1, \cdots, m_n$ such that $b\prod_{i=0}^{n} m_i$ converges to a definite limit k (to a desired significance or accuracy) for some reasonable small n. The dividend a is also simultaneously multiplied by the same sequence of multipliers m_i. In other words, denoting $a = y_0$ and $b = x_0$, the procedure consists in implementing the pairs of recursions

$$x_{i+1} = m_i x_i \quad \text{and} \quad y_{i+1} = m_i y_i \tag{1}$$

so that for a reasonable i

$$x_i \to k \quad \text{and} \quad y_i \to kq$$

so that

$$q = y_i k^{-1}.$$

Thus the procedure requires selection of m_i and multiplications and a final step to multiply by k^{-1}. It is therefore desirable that m_i is chosen to be easily computable, and at the same time k^{-1} is a convenient integer so that $k^{-1} \cdot y$ is preferably obtained using only shift operations. In fact it is easy to see that this method turns out to be more general than the Wilkes–Harvard scheme where $k = 1$, so that $q = y_i$.

The advantage of convergence to k and the recursions that would be needed for the purpose will be discussed in the succeeding sections.

The Choice of Limits of Convergence and Recursions

Although various iterative methods can be devised to enable a given sequence to converge to k, the form we desire is the multiplicative form of recursion (2) based on the classical functional iterative scheme [11], [7]:

$$x_{i+1} = x_i \cdot m_i = \phi(x_i). \tag{2}$$

Starting with $x_0 = b$ it is necessary that (2) converges to the root k of a polynomial $f(x)$ given by

$$f(x) = \phi(x) - x = 0.$$

It is known that the degree of convergence depends on the degree of $f(x)$ as well as on the roots of $f(x)$. Our interest will be confined to the case of quadratic convergence as this appears to be more convenient for hardware realization. Therefore, it is sufficient to take k as a root of the quadratic

$$f(x) = (x - k_1)(x - k_2) = 0 \tag{3}$$

with $k_1 = k$, and derive a functional iterative process of the form (2) to converge to its root k. (More general higher order processes can be constructed on similar principles.)

From (3) we rearrange and see that one such iterative process is of the form

$$x_{i+1} = x_i \cdot \frac{(k_1 + k_2 - x_i)}{k_2} = \phi(x_i). \tag{4a}$$

It is known from standard books on numerical analysis [11] that the convergence of (4a) at a point is fastest if the derivative $\phi'(x)$ is a minimum or zero in the neighborhood of $x = k = k_1$.

Therefore, from (4a) the fastest convergence is obtained when

$$\phi'(x) = -\frac{2x}{k_2} + \frac{k_1 + k_2}{k_2} = 0 \qquad \text{at } x = k,$$

which means the quadratic $f(x)$ should have two equal roots:

$$k_1 = k_2 = k.$$

Using this we obtain the best form of iteration as

$$x_{i+1} = \frac{x_i}{k} \cdot (2k - x_i) = x_i \cdot m_i \tag{4b}$$

where

$$m_i = \frac{(2k - x_i)}{k}.$$

Thus

$$y_{i+1} = y_i \cdot m_i \tag{4c}$$

is the form of recursion for the dividend to converge to kq. The recursion (4b) converges to k provided $0 < x_i < 2k$. (Note that $(x_i/k)(2k - x_i)$ is monotonic in the intervals $0 < x < k$ and $k < x < 2k$.) For $k = 1$, (4b) reduces to the Wilkes–Harvard scheme.

It is worthwhile noting that while the iterative process (4b) is self-correcting for convergence to k within the range $0 < x_i < 2k$, the sequence (4c) has to be carried out without any error in y_i. In this sense (4b) is an iterative process while (4c) is a mere recursive process using the exact values of y_i and the same values of m_i used for (4b).

Practical Choices for the Limits of Convergence

We will now consider the choices of k which would give rise to simple forms of recursions. We will consider only radices β which are powers of two as these are of practical interest, although the method is equally applicable to other radices.

It turns out that the choices $k = 1/2^s$ for $s = 0, 1, 2, \cdots$ are suitable for implementation with multiply, shift, and complement operations (the choice of $k = 2/3$ seems practical with an extra add operation). If we denote a left bit shift of x_i (or y_i) by $L(x_i)$ (or $L(y_i)$) and denote $L^r(x)$ (or $L^r(y)$) as r left shifts or right shifts accordingly as r is positive or negative, respectively, and denote

$$N(x) = (1 - x) = \text{one's complement of } x,$$

we obtain from (4b) and (4c) the following recursions for a given $k = 1, 1/2, 1/2^{r+1}$, respectively.

$\underline{k = 1}$

$$x_{i+1} = L(x_i \cdot N(L^{-1}(x_i))); \quad y_{i+1} = L(y_i \cdot N(L^{-1}(x_i))) \tag{5a}$$

$\underline{k = \frac{1}{2}}$

$$x_{i+1} = L(x_i \cdot N(x_i)); \qquad y_{i+1} = L(y_i \cdot N(x_i)) \tag{5b}$$

$\underline{k = \dfrac{1}{2^{r+1}}}$

$$x_{i+1} = L(x_i \cdot N(L^r(x_i))); \quad y_{i+1} = L(x_i \cdot N(L^r(x_i))) \tag{5c}$$

Note that (5a) is the same as the Wilkes–Harvard scheme implemented by Lehman [1] and Anderson *et al.* [5].

However, except for (5a) and (5b), the general scheme (5c) is of no practical utility for normalized divisors.

If one chooses $k = 2/3$, we obtain

$$x_{i+1} = L(x_i \cdot (N(x_i) + L^{-2}(x_i)))$$

and

$$y_{i+1} = L(y_i \cdot (N(x_i) + L^{-2}(x_i))) \tag{5d}$$

where an extra addition is involved.

Before we discuss the merits of these schemes we will devise some measures for the efficiency of an iterative scheme.

A Measure for the Efficiency of Iterative Schemes

A convenient and a reasonably accurate measure is the determination of number of significant digits in the nth iterate x_n in representing k. (Then the corresponding significance is obtained for the quotient.) Let us denote this by $\sigma(x_n)$. This is very closely related to the

logarithm (to base β) of the reciprocal of the relative error of x_n with respect to k. (See [8].)

Thus

$$\sigma(x_n) \approx \log_\beta \frac{k}{|k - x_n|}. \tag{6}$$

Using (4b) we obtain

$$x_n = k - k\left(1 - \frac{x_0}{k}\right)^{2^n}.$$

Thus

$$\sigma(x_n) \approx 2^n \log_\beta \left(\frac{k}{|k - x_0|}\right). \tag{7}$$

For the significance $\sigma(x_n)$ to be at least σ', a preassigned level, the choice of n is determined by the inequality

$$2^n \log_\beta \frac{k}{|k - x_0|} \geq \sigma'. \tag{8}$$

For example, for $\sigma' = 16$, $\beta = 2$, $x_0 \approx 1/2$, $k = 1$ we need

$$2^n \log_2 2 \geq 16 \quad \text{or} \quad n \geq 4.$$

Choice of Convergent Limits and Speed of Convergence

We will now use the measure outlined above to obtain the speed of convergence and arrive at optimal schemes.

It is well known [7] that the speed of convergence depends on the derivative $\phi'(x)$ in (4a). By obtaining the derivatives $\phi'(x_i)$ for $k = 1/2$ and $k = 1$, it is easily shown that for $x_0 = 2/3$ the speed of convergence to $k = 1/2$ using (5b) is the same as that for $k = 1$ using (5a). Thus it is necessary to choose the recursions (5b) or (5a) according to $\frac{1}{2} \leq x < 2/3$ or $2/3 \leq x \leq 1$, respectively. This would result in equal speeds of convergence. (One can use (7) to prove this directly.)

From (8) we see that the worst case arises for $x_0 = 2/3$, $k = 1$, $\sigma' = 32$, $n = 5$. Also by simple computation, it is seen that except for $21/32 \leq x_0 \leq 23/32$ which are close to $2/3$, the selection of (5b) or (5a) accordingly as $1/2 \leq x < 21/32$ or $21/32 \leq x < 1$ results in a speed the same as in Lehman's implementation [1], namely, the convergence is obtained in four iterations with 32-bit accuracy.

If, however, we introduce an additional recursion (5d) so that we can converge to $2/3(0.101010..)$ so $k^{-1} = 3/2$ and $k^{-1}y_i$ is easily computed as shifts and addition, then by obtaining $\phi'(x_i)$ for $k = 2/3$ it is shown that the speed of convergence for the schemes (5a), (5b), and (5d) will be equal accordingly as x_0 satisfies the following ranges.

$$\begin{aligned} &\textit{Scheme (5b)}: && 1/2 \leq x_0 < 4/7 (\approx 9/16) \\ &\textit{Scheme (5d)}: && 4/7 \leq x_0 < 4/5 (\approx 13/16) \\ &\textit{Scheme (5a)}: && 4/5 \leq x_0 < 1. \end{aligned} \tag{9}$$

Using (8) we see that with $n = 4$, we get at least 36-bit accuracy for these choices, which is a little faster than the table-lookup implementation [1]. It is, however, to be noted that the gain is only one iteration in introducing an additional convergence limit at $k = 2/3$.

This means, for a given accuracy or significance and speed of convergence, the present schemes offer the possibility of reduction in the size of the table used in [1] and [5] by using a more complex control to choose one of the appropriate schemes according to the range (9) in which the divisor lies.

Precision (Working Accuracy) Needed for the Multiplier

We have been assuming so far that the recursions (4b) and (4c) are carried out to p-precision, namely, x_i, y_i, and m_i are all p-precision numbers. It is, however, possible to reduce the precision of $m_i = (2k - x_i)/k$, particularly during the early stages of iteration. This would reduce the computational labor and would speed up the scheme, provided simple additional tests are carried out. It will be shown below that the length of m_i can be considerably smaller than p, depending upon the absolute error in x_i.

It is seen that when m_i is of full-precision p, the recursion (4b) converges to k monotonically. By using a truncated x_i (denoted by x_{iT}) to form $m_{iT} = (2k - x_{iT})/k$, the convergence to k can become either oscillatory or monotonic according to the absolute error in x_{iT}. This will be explained below.

Case 1 (Oscillatory Convergence): Let $x_i = k \pm \delta$ where δ is the absolute error in x_i. If we assume that we truncate x_i to j digits (x_{iT}), then

$$x_{iT} = x_i - \epsilon$$

where

$$0 \leq \epsilon < \beta^{-j}.$$

Thus

$$m_{iT} = (2k - x_i + \epsilon)/k.$$

If we use m_{iT} instead of m_i in (4b) to compute x_{i+1} (note that x_i used is of full-precision p),

$$x_{i+1} = m_{iT} \cdot x_i \geq m_i \cdot x_i,$$

then x_{i+1} could become larger than k even if $x_i < k$, and the sequence becomes oscillatory.

If we desire that the relative error still decreases at the same geometric rate in spite of this error in m_{iT}, we should insist that for the sequence (4b)

$$|(k - x_{i+1})/k| = ((k - x_i)/k)^2$$

or

$$|k - x_{i+1}| \leq \delta^2/k$$

or

$$(k - \delta^2/k) \leq x_{i+1} \leq k + \delta^2/k \tag{10a}$$

where

$$x_{i+1} = ((k \pm \delta)/k)(k \mp \delta + \epsilon). \quad (10b)$$

From (10a) we obtain that ϵ the truncation error in x_{iT} must satisfy

$$0 \leq \epsilon \leq 2\delta^2/(k + \delta) \quad (11)$$

or

$$\beta^{-j} \leq 2\delta^2/(k + \delta)$$

or

$$j \geq \log_\beta k - 2 \log_\beta \delta - \log_\beta 2 + \log_\beta (1 + \delta/k). \quad (12a)$$

For $k \leq 1$, $\delta \ll k$, (12a) can be approximated to

$$j \geq -2 \log_\beta \delta - \log_\beta 2. \quad (12b)$$

Also since δ is the absolute error in x_i, it can be expressed as

$$1/\beta^{l+1} < \delta \leq 1/\beta^l. \quad (13)$$

Using (13) we obtain from (12b)

$$j \geq 2l + 2 - \log_\beta 2 \quad (12c)$$

or

$$j \geq 2l + 1 \qquad (\beta = 2). \quad (12d)$$

Note that (13) actually denotes, for $k = 1$, that x_i has either a leading digit 1 followed by at least l zeros or has l leading repeated digits of magnitude $(\beta - 1)$. This serves as a test to select the precision of m_i. For example, using (12d) for $k = 1/2$, $p = 48$, $\beta = 2$, $1/2^5 < \delta \leq 1/2^4$, we find that $j \geq 9$. This means it is sufficient to use only the most significant 9 bits of x_i (denoted as x_{iT}) to form $2(1 - x_{iT}) = m_{iT}$ to multiply x_i and obtain x_{i+1}. Then x_{i+1} will have at least eight leading 1's or a leading 1 followed by eight zeros, keeping up the same rate of convergence that would otherwise have been achieved using m_i (see [5]).

Case 2 (Monotonic Convergence): If in spite of using m_{iT} rather than m_i to compute x_{i+1}, we desire that the iterates converge monotonically to k, a little more precision is needed for m_{iT} than desired by (12d). In some cases this may be desirable if overflow is to be eliminated.

We find for this case that instead of (10a) the following inequality has to be satisfied:

$$(k - \delta^2/k) \leq x_{i+1} \leq k. \quad (14)$$

(Note that even if $x_0 = k + \delta$ to start with, it is possible to converge from the lower side since $x_1 < k$; therefore, we set $x_i = (k - \delta)$ in this argument.)

As before we obtain from (14)

$$0 \leq \epsilon \leq \delta^2/(k - \delta)$$

or

$$\beta^{-j} \leq \delta^2/(k - \delta)$$

or

$$j \geq \log_\beta k - 2 \log_\beta \delta + \log_\beta (1 - \delta/k). \quad (15a)$$

For $k \leq 1$, $\delta \ll k$, and $1/\beta^{l+1} < \delta \leq 1/\beta^l$ we obtain from (15a)

$$\geq 2l + 2. \quad (15b)$$

This means, for the example considered, namely $k = 1/2$, $p = 48$, $\beta = 2$, $1/2^5 < \delta \leq 1/2^4$, it is necessary that $j \geq 10$ or it is sufficient to use only the most significant 10 bits of x_i (denoted as x_{iT}) to form $2(1 - x_{iT}) = m_{iT}$, to multiply x_i, and obtain x_{i+1}. Then x_{i+1} will have at least eight leading 1's and would be less than k.

Of course, this economy is possible only as long as $(2l+2) \leq p$; otherwise one has to work with a p-precision m_i. Thus this economy can be achieved as long as δ is not very small, which is so at the early stages of iteration. (See [5].)

Optimal Number of Multipliers for the Range Transformation of Divisor

Our discussion so far was confined to the modified iterative methods. We will now show how the results obtained above could be used to arrive at an optimal scheme for systematically selecting the multipliers, if one prefers table-lookup scheme [1], [5] and converge to $k = 1$ (scheme (5a)). We will illustrate this in a practical case.

Assume that we need a 48-bit quotient by using scheme (4a) in, say, four iterations. Then from (8) we obtain

$$2^4 \log_2 k/|k - x_0| = 48 \quad \text{or} \quad k/(k - x_0) = 8. \quad (16)$$

This means the choice of each converging limit k_i and the initial value x_{oi} in the interval $1/2 \leq x < 1$ should obey (16) for the optimal scheme. If we now construct a sequence

$$k_1, x_{01}, k_2, x_{02} \cdots 1$$

where

$$k_1 < x_{01} < k_2 < x_{02} < k_3 < x_{03} \cdots 1$$

by alternatively placing the convergence limits and initial values of x until $k_i = 1$, then to obey (16), starting with $k_1 = 1/2$, we have in this sequence

$$x_{oi} = (9/8)k_i \qquad (x_{oi} > k_i) \quad (17)$$

and

$$k_{i+1} = (8/7)x_{oi} \qquad (k_{i+1} > x_{oi}). \quad (18)$$

This sequence (which may be called the characteristic sequence) for the above practical case is

$$1/2, 9/16, 9/14, 81/112, 81/98, 729/784, 1.$$

From this, one can either choose a point of convergence for the given starting values (i.e., x_{oi} can converge to either k_i or k_{i+1}), or equivalently one could transform the numbers in the intervals $k_i \leq x < x_{oi}$ and $x_{oi} \leq x$

TABLE I

Range of Divisor	Multiplier	Transformed Range
$1/2 \leq x < 9/16$	2 16/9	$1 \leq x < 9/8$ $8/9 \leq x < 1$
$9/16 \leq x < 9/14$	16/9 14/9	$1 \leq x < 8/7$ $7/8 \leq x < 1$
$9/14 \leq x < 81/112$	14/9 112/81	$1 \leq x < 9/8$ $8/9 \leq x < 1$
$81/112 \leq x < 81/98$	112/81 98/81	$1 \leq x < 8/7$ $7/8 \leq x < 1$
$81/98 \leq x < 729/784$	98/81 784/729	$1 \leq x < 9/8$ $8/9 \leq x < 1$
$729/784 \leq x < 1$	...	No transformation is required.

$< k_{i+1}$ to a range close to unity (above and below) by multiplication with an approximation to the reciprocal of k_i or x_{oi} and converge to 1. This gives us an optimal scheme for selecting the multipliers for a given speed of convergence and significance.

In Table I the range of divisor, selected multipliers, and the transformed range are given. From this table it is seen that we need the set of multipliers 2 or 14/9 or 98/81 accordingly as $1/2 \leq x < 9/16$ or $9/16 \leq x < 81/112$ or $81/112 \leq x < 729/784$, respectively.

Also note that the sequence (1/2, 9/16, 9/14, 81/112, 81/98, 729/784, 1) can be replaced by the approximating 6-bit representations (1/2, 36/64, 41/64, 46/64, 52/64, 56/64, 1) without loss in speed; and the multipliers 2, 64/41, and 64/52 could also be replaced by the approximating representations of 2, 25/16, and 5/4, namely, 10.00, 1.1001, and 1.01, respectively. This would help minimize the cost of table-lookup procedure with no loss in speed.

In general, for obtaining a p-precision quotient in n-iterations, the minimal number of multipliers needed is obtained as follows.

Let $k/|k - x_0|$ $\text{antilog}_2(p/2^n) = \gamma$. Using a similar argument, we obtain relations similar to (17) and (18) for x_{0i} and k_i. From these, the minimal number of multipliers needed for the transformation is given by minimum (i_1, i_2) where i_1 and i_2 satisfy the inequalities

$$i_1 \log_2 ((\gamma + 1)/(\gamma - 1)) + \log_2 k_1 \geq 0 \qquad (19)$$

and

$$i_2 \log_2 ((\gamma + 1)/(\gamma - 1)) + \log_2 x_{01} \geq 0. \qquad (20)$$

It is, however, necessary to remark that while the papers of Lehman [1] or Anderson [5] do not contain any systematic method of constructing the optimal multiplier table, it seems that the tables presented have been constructed in the economical manner for the given accuracy and speed.

Acknowledgment

The author wishes to express his gratitude to the Wiezmann Institute of Science for their hospitality and support during the course of this work, and to the referees for their constructive remarks.

References

[1] M. Lehman, D. Senzig, and J. Lee, "Serial arithmetic techniques," *1965 Fall Joint Computer Conf., AFIPS Proc.*, vol. 27, pt. 1. Washington, D. C.: Spartan, 1965, pp. 715–725.
[2] D. Ferrari, "A division method using a parallel multiplier," *IEEE Trans. Electronic Computers*, vol. EC-16, pp. 224–226, April 1967.
[3] C. S. Wallace, "A suggestion for a fast-multiplier," *IEEE Trans. Electronic Computers*, vol. EC-13, pp. 14–17, February 1964.
[4] M. J. Flynn, "Very high speed computing systems," *Proc. IEEE*, vol. 54, pp. 1901–1909, December 1966.
[5] S. F. Anderson, J. G. Earle, R. E. Goldschmidt, and D. M. Powers, "The IBM system/360 model 91, floating point execution unit," *IBM J. Res. and Develop.*, vol. 11, pp. 34–53, January 1967.
[6] R. K. Richards, *Arithmetic Operations in Digital Computers*. London: Van Nostrand, 1955.
[7] A. A. Goldstein, *Constructive Real Analysis*. New York: Harper and Row, 1967.
[8] G. W. Evans, II, and C. L. Perry, *Programming and Coding for Automatic Digital Computers*. New York: McGraw-Hill, 1961, p. 207.
[9] M. V. Wilkes, D. J. Wheeler, and S. Gill, *The Preparation of Programs for an Electronic Digital Computer*. Cambridge, Mass.: Addison-Wesley, 1951.
[10] R. E. Goldschmidt, "Applications of division by convergence," M.S. thesis, Massachusetts Institute of Technology, Cambridge, Mass., June 1964.
[11] J. B. Scarborough, *Numerical Mathematical Analysis*. London: Oxford University Press, 1955, pp. 201–203.