A HIGHER-RADIX DIVISION WITH SIMPLE SELECTION OF QUOTIENT DIGITS

Miloš D. Ercegovac

Reprinted from PROCEEDINGS OF IEEE 1983
6TH SYMPOSIUM ON COMPUTER ARITHMETIC

A HIGHER-RADIX DIVISION WITH SIMPLE SELECTION OF QUOTIENT DIGITS

Miloš D. Ercegovac
UCLA Computer Science Department
University of California, Los Angeles

ABSTRACT

A higher-radix division algorithm with simple selection of quotient digits is described. The proposed scheme is a combination of the multiplicative normalization used in the continued-product algorithms and the recursive division algorithm. The scheme consists of two parts: in the first part, the divisor and the dividend are transformed into the range which allows the quotient digits to be selected by rounding partial remainders to the most significant radix-r digit in the second part. Since the selection requires only the most significant part of the partial remainder, limited carry-propagation adders can be used to form the partial remainders. The divisor and dividend transformations are performed in three steps using multipliers of the form $1 + s_k r^{-k}$ as in the continued product algorithm. The higher radix of the form $r = 2^k$, k=2,4,8,.... can be used to reduce the number of steps while retaining the simple quotient selection rules.

I. INTRODUCTION

In this article a division scheme characterized by a simple method for selecting quotient digits is described. The scheme also has several properties important for modular implementation. Division algorithms have been of a wide interest [ROBE58, METZ62, ATKI68, ANDE68, TAYL81] because of the problems: (i) fast and efficient selection of quotient digits, (ii) computation of partial remainders, and (iii) compatibility of implementation with other more frequent arithmetic operations such as multiplication.

The scheme for division suggested here consists of two parts. In the first part the divisor X is forced into a suitable range and the dividend Y is adjusted. The divisor and dividend transformations are performed using a few initial steps of the iterative multiplicative normalization algorithm [ERCE73, DELU70, ROBE73]. In the second part the quotient digits are obtained by a recursive algorithm [ERCE75, ERCE77] in which the selection can be performed by rounding. The proposed division scheme generates an m-digit quotient in m+3 additive steps which do not require full precision carry propagation. The scheme also provides the remainder.

The division schemes based on the range transformation have been considered before [SVOB63, KRIS70, ERCE75]. The main contributions of this article are implementation-efficient range transformation and a simple quotient selection method which does not depend on the radix.

In Section II a derivation of the division scheme is presented. A radix-16 division algorithm is given in Section III. The implementation aspects are discussed in [ERCE83].

II. DERIVATION OF THE DIVISION SCHEME

Consider the division problem

$$Y = XQ + R \tag{1}$$

where

X is the n-bit divisor, $|X| \in [1/2, 1)$;

Y is the 2n-bit dividend, $|Y| < |X|$;

Q is the n-bit quotient and

R is the corresponding remainder.

A binary recursive division algorithm computes sequentially the partial remainders and the quotient digits using the recursion

$$R_{j+1} = 2R_j - q_{j+1}X, \quad j=0,1,2,...,n-1 \tag{2}$$

where

$R_0 = Y$ is the initial remainder,

$q_{j+1} = SELECT(R_j, X)$ is the j+1-th quotient bit, and

$SELECT$ is a selection function.

In order to reduce the number of steps, the binary algorithm can be modified so that b bits of the quotient are obtained per step. That is, the radix of implementation is defined to be $r = 2^b$. However, the use of a higher radix makes the selection of the quotient digits as well as the computation of the partial remainders more complex [ROBE58, ATKI68]. The computation of partial remainders can be simplified by precomputing necessary multiples of the divisor so that the recursion step takes about the same time as in the binary case. However, the selection of higher radix quotient digits remains difficult. In this paper we describe a higher radix division algorithm in which (i) the selection can be performed by a simple rounding and (ii) advantages of higher radix can be exploited.

A. Recursion and Selection

The recursive algorithm for division in which the quotient digits are obtained by rounding partial remainders to the integer part and taking the integer part as the quotient digit requires the divisor to be in the range

$$[1-\alpha,\ 1+\alpha] \tag{3}$$

where α is a constant between 0 and 1, to be determined later. It also requires the use of a redundant representation of the quotient digits. A symmetric redundant digit set (signed-digit set [AVIZ61]) is used:

$$D_\rho = \{-\rho,\ldots,-1,0,1,\ldots,\rho\} \tag{4}$$

where for radix r

$$r/2 \leq \rho < r$$

The recursion is

$$R_j = r(R_{j-1} - q_{j-1}X^*) \tag{5}$$

and

$$q_j = SELECT(R_j) \tag{6}$$

$$= \begin{cases} sign\ R_j \left\lfloor |R_j| + 1/2 \right\rfloor & \text{if } |R_j| \leq \rho \\ sign\ R_j \left\lfloor |R_j| \right\rfloor & otherwise \end{cases}$$

where

R_j *is the j−th remainder;*

X^* *is the scaled divisor such that*

$$1-\alpha \leq |X^*| \leq 1+\alpha,$$

and

$q_j \in D_\rho$ *is the j−th quotient digit.*

Initially,

$$R_0 = Y^*$$

is the scaled dividend Y such that

$$|R_0| \leq \rho + \beta \text{ and } 1/2 \leq \beta < \frac{\rho}{r-1} \tag{7}$$

The validity of the recursion and the selection function is established by proving the following two claims.

Claim 1:

If the bound α is

$$0 < \alpha \leq \frac{1}{r}\left[1 - \frac{\beta(r-1)}{\rho}\right] \tag{8}$$

and q_j is the j-th quotient digit from a signed-digit set $D_\rho = \{-\rho,\ldots,-1,0,1,\ldots,\rho\}$, $r/2 \leq \rho < r$, selected according to the function *SELECT*, then the partial remainder R_j satisfies

$$|R_j| \leq \rho + \beta \tag{9}$$

for all j.

Proof:

To show that the partial remainders are bounded we proceed by induction. By definition (7):

$$|R_0| \leq \rho + \beta$$

Assume

$$|R_{j-1}| \leq \rho + \beta$$

Let $A = 1 - X^*$ so that $|A| = \alpha$. Then

$$|R_j| \leq r|R_{j-1} - q_{j-1}| + r|A||q_{j-1}| \tag{10}$$

$$\leq r(\rho + \beta - \rho) + r\alpha\rho$$

$$= r\beta + r\left[\frac{1}{r}\left(1 - \frac{\beta(r-1)}{\rho}\right)\right]\rho$$

$$= \rho + \beta$$

because, by definition of the selection function *SELECT*, the choice of digit q_j can always be made such that

$$|R_j - q_j| \leq \beta \tag{11}$$

□

Claim 2:

Let $Q^* = \sum_{i=0}^{m} q_i r^{-i}$ be the computed quotient. Then

$$\left|\frac{Y^*}{X^*} - Q^*\right| \leq r^{-m} \tag{12}$$

Also, $R = r^{-m-1}R_{m+1}$.

Proof:

By substitution

$$Y^* = X^*\sum_{i=0}^{m} q_i r^{-i} + r^{-m-1}R_{m+1} \tag{13}$$

and

$$\left|\frac{Y^*}{X^*} - Q^*\right| \leqslant r^{-m-1} \frac{|R_{m+1}|_{\max}}{|X^*|_{\min}} \tag{14}$$

$$= r^{-m-1}\left[\frac{\rho + \beta}{1 - \alpha}\right]$$

$$= r^{-m} \quad \text{for } \rho = r-1$$

$$< r^{-m} \quad \text{for } \rho < r-1$$

From (13, 14), $R = r^{-m-1}R_{m+1}$.

□

According to the analysis of the rounding selection method [ERCE75] the bounds α, β, ρ and the selection interval overlap Δ are related as follows. First, in order to have efficient implementation of single-digit multipliers, required by the division recursion, the maximum digit value should be [ATKI70]:

$$\rho \leqslant \frac{2(r-1)}{3}. \tag{15}$$

Therefore, from (7):

$$1/2 < \beta < 2/3$$

On the other hand, $\beta = \frac{1}{2}(1 + \Delta)$, where Δ is the overlap between the selection intervals [ERCE75]. Therefore, the upper bound on α can be written as:

$$\alpha \leqslant \frac{1}{r}[1 - \frac{3\beta}{2}]$$

$$= \frac{1}{r}[1 - \frac{3(1+\Delta)}{4}]$$

For $\Delta = 1/r$,

$$\alpha \leqslant \frac{r-3}{4r^2} \tag{16}$$

This bound will be used to define the range of the transformed divisor.

To transform the divisor into this range and adjust the dividend Y, we adopt the multiplicative normalization technique [DELU70, ERCE73].

B. Range Transformation

The multiplicative normalization of a given argument

$$|X_0| \in [1/2, 1)$$

is performed as a sequence of transformations such that

$$1-\alpha \leqslant |X_0 \prod_{i=0}^{p-1} M_i| \leqslant 1+\alpha \tag{17}$$

for a given constant $0 \leqslant \alpha < 1$ and the number of steps p. The multipliers are of the form $M_k = 1 + S_k r^{-k}$, where r is the radix and S_k is a digit in a redundant radix r number system. The form of the multipliers simplifies the implementation since the full-precision multiplication is replaced by an addition, a single radix-r digit multiplication and a k-position shift.

The multiplicative normalization is performed recursively:

$$X_{k+1} = X_k(1 + S_k r^{-k}), \ 0 \leqslant k < p \tag{18}$$

The digit value of S_k is chosen such that the error e_{k+1} after step k is

$$|e_{k+1}| = |1 - X_k(1 + S_k r^{-k})| \leqslant \frac{\rho}{r-1} r^{-k} \tag{19}$$

The number of the transformation steps p can now be obtained from the following condition, implied by (16) and (19):

$$|e_p| \leqslant \alpha \tag{20}$$

Assuming that the overlap $\Delta = \frac{1}{r}$ and $r \geqslant 8$, it follows that $p \geqslant 3$. That is, three steps are sufficient to transform given divisor X and dividend Y into the required range.

The multiplicative normalization is conveniently performed using a recursion on scaled differences (remainders). Let

$$D_k = r^{k-1}(X_k - 1), \quad 0 < k < p \tag{22}$$

From (18) and (22), the scaled difference recursion follows:

$$D_{k+1} = rD_k + S_k + S_k D_k r^{-k+1}, \quad 0 < k < p \tag{23}$$

For p=3, the normalization procedure requires determination of S_0, S_1 and S_2. A complete derivation procedure for the selection rules is discussed in [ERCE72]. For the sake of brevity, we only show the radix-16 rules in the next section.

III. RADIX-16 ALGORITHM

In this section the division scheme is illustrated for r=16. The algorithm is as follows:

```
begin
/* Part 1 - Range Transformation
/* Inputs:  Divisor X0 ∈ [1/2,1)
/*          Dividend Y0, |Y0|<|X0|
/* Outputs: Transformed divisor X*
/*          Transformed dividend Y*

1: if 1/2≤X0<5/8
     then
       D1 ← 2X0 − 1
       Y1 ← 2Y0
     else
       D1 ← X0 − 1
       Y1 ← Y0
```

2: $S_1 \leftarrow \overline{sign} D_1 \left\lfloor 16(D_1 + U_1) \right\rfloor$
3: $D_2 \leftarrow 16D_1 + S_1 + S_1 D_1$
$Y_2 \leftarrow Y_1(1 + S_1 16^{-1})$
4: $S_2 \leftarrow \overline{sign} D_2 \left\lfloor 16(D_2 + U_2) \right\rfloor$
5: $X^* \leftarrow (16D_2 + S_2 + S_2 D_2 16^{-1} + 1)16^{-2}$
$Y^* \leftarrow Y_2(1 + S_2 16^{-2})$
$q_{-1} = 0$

/* **Part 2 - Division Recursion**
/* Inputs: Divisor X^*
/* Dividend Y^*
/* Outputs: Quotient $Q^* = \sum_{i=0}^{m} q_i 16^{-i}$
/* Remainder $R = 16^{-m-1} R_{m+1}$

7: **for** j = 0,1,2,..., m **do**
7.1: $R_j \leftarrow 16(R_{j-1} - X^* q_{j-1})$
7.2: $q_j \leftarrow SELECT(\hat{R}_j)$
end

The selection function *SELECT*, defined in (6), is performed on an estimate $\hat{R}_j$ of the partial remainder such that $|\hat{R}_j - R_j| \leq \frac{1}{16}$. The terms U_1 and U_2 are six-bit rounding constants defined as functions of the seven leading bits of the truncated scaled difference D_j, j=0,1,2.

$$U_k = \sum_{i=1}^{6} u_i 2^{-i} \tag{24}$$

where the switching expressions for u_i's are

$$u_1 = u_2 = 0,$$

$$u_3 = K1\, d_0 \bar{d}_2,$$

$$u_4 = K1\, d_0 \bar{d}_4 (\bar{d}_2 + \bar{d}_3),$$

$$u_5 = K1(d_0 + \bar{d}_3 \bar{d}_4) + K2[d_0 + \bar{d}_1(\bar{d}_2 + \bar{d}_3) + d_6]$$

$$u_6 = K1\, \bar{d}_3 d_4 + K2\, d_0(d_1 + d_2 d_3)$$

and K1 and K2 denote steps 1 and 2, respectively. The derivation of these step-dependent rounding constants is based on the selection intervals given in the Appendix. A more detailed discussion can be found in [ERCE72]. An example of the division algorithm is given in Figure 1.

IV. CONCLUSION

A scheme for higher radix division has been presented. It consists of a 3-step transformation of the divisor and the dividend into a range which allows use of a recursive higher radix division algorithm with a simple quotient selection method. A detailed derivation of the range transformation requirements and the procedure has been described and an algorithm for r=16 has been given. The implementation details and the performance are discussed elsewhere [ERCE83].

Divisor $X_0 = 0.8107509300$,
Dividend $Y_0 = 0.5990471500$,
Quotient $Q = 0.7388793868$

Part 1:

After Step 1: $D_1 = -0.1892490700$, $Y_1 = 0.5990471500$, $S_1 = 4$
After Step 2: $D_2 = 0.2150186000$, $Y_2 = 0.7488089375$, $S_2 = -3$

Transformed divisor and dividend:

$X^* = 1.0015624282$, $Y^* = 0.7400338328$

Part 2:

i	Remainder	q	Quotient	Error
1	-4.1844575266	1	1.0000000000	-0.2611206132
2	-2.8513250219	-4	0.7500000000	-0.0111206132
3	2.4537962023	-3	0.7382812500	0.0005981368
4	7.2107415359	2	0.7387695313	0.0001098555
5	3.1968726195	7	0.7388763428	0.0000030440
6	3.0749653602	3	0.7388792038	0.0000001830
7	1.1244492114	3	0.7388793826	0.0000000042
8	1.9661885316	1	0.7388793863	0.0000000005

(All numbers are represented in decimal)

Figure 1: Example of Division

REFERENCES

[ANDE68] S.F. Anderson, J.G. Earle, R.E. Goldschmidt, and D.M. Powers, "The IBM/360 Model 91: Floating-Point Execution Unit", IBM Journal, January 1967, pp.34-53.

[AVIZ61] A. Avizienis, "Signed Digit Number Representations for Fast Parallel Arithmetic", IRE Trans. on Electronic Computers, 1961, pp.389-400.

[ATKI68] D. E. Atkins, "Higher-Radix Division Using Estimates of the Divisor and Partial Remainders", IEEE Trans. on Computers, Vol.C-17, No. 10, October 1968, pp.925-934.

[ATKI70] D. E. Atkins, "Design of the Arithmetic Units of ILLIAC III: Use of Redundancy and Higher Radix Methods", IEEE Trans. on Computers, August 1970, pp.720-733.

[DELU70] B.G. DeLugish, " Class of Algorithms for Automatic Evaluation of Certain Elementary Functions in a Binary Computer", Ph.D. Dissertation, University of Illinois at Urbana-Champaign, Department of Computer Science, June 1970.

[ERCE72] M.D. Ercegovac, "Radix 16 Division, Multiplication, Logarithmic and Exponential Algorithms Based on Continued Product Representations", MS Thesis, University of Illinois at Urbana-Champaign, Department of Computer Science, August 1972.

[ERCE73] M. D. Ercegovac, "Radix-16 Evaluation of Certain Elementary Functions", IEEE Trans. on Computers, June 1973, pp.561-566.

[ERCE75] M.D. Ercegovac, "A General Method for Evaluation of Functions and Computations in a Digital Computer", Ph.D. Dissertation, University of Illinois at Urbana-Champaign, Department of Computer Science, August 1975.

[ERCE77] M. D. Ercegovac, "A General Hardware-Oriented Method for Evaluation of Functions and Computations in a Digital Computer", IEEE Trans. on Computers Vol. C-26, No.7, July 1977, 667-680.

[ERCE83] M.D. Ercegovac, "Higher-Radix Division: Implementation Alternatives and Their Performance", (in preparation), 1983.

[KRIS70] E.V. Krishnamurthy, "On Range-Transformation Techniques for Division", IEEE Transactions on Computers, Vol.C-19, No.3, March 1970, pp.227-231.

[METZ62] G. Metze, "A Class of Binary Divisions Yielding Minimally Represented Quotients", IRE Trans. on Electron. Computers, Vol.EC-11, No.6, December 1962, pp.761-764.

[ROBE58] J. E. Robertson, "A New Class of Digital Division Methods", IRE Trans. on Electronic Computers, September 1958 pp. 88-92.

[ROBE73] J.E. Robertson and K.S. Trivedi, "The Status of Investigations into Computer Hardware Design Based on the Use of Continued Fractions", IEEE Trans. on Computers, Vol.C-22, No.6, June 1973, pp.555-560.

[SVOB63] A. Svoboda, "An Algorithm for Division", Information Processing Machines, Vol.9, 1963, pp.25-32.

[TAYL81a] G. S. Taylor, "Compatible Hardware for Division and Square Root", Proc. 5th Symposium on Computer Arithmetic, 1981, pp.127-134.

[TUNG68] C. Tung, "A Division Algorithm for Signed-Digit Arithmetic", IEEE Transactions on Computers, Vol.17, 1968, pp.887-889.

[ZURA81] J. H. P. Zurawski and J. B. Gosling, "Design of High-Speed Digital Divider Units", IEEE Trans. on Computers, Vol. C-30, No.9, September 1981, pp.691-699.

Acknowledgements: This work has been supported in part by a research grant from the Batelle Memorial Institute and by the ONR contract N00014-79-C-0866 (Research in Distributed Processing).

Appendix

The selection intervals for S_1 and S_2 are shown in Tables 1 and 2, respectively. The detailed procedure for the derivation is given in [ERCE72].

S_1	$64\underline{D}_1$	$64\overline{D}_1$
10	-26	-23
9	-24	-22
8	-23	-20
7	-21	-18
6	-19	-16
5	-17	-14
4	-14	-11
3	-12	-8
2	-9	-5
1	-6	-2
0	-2	3
-1	2	7
-2	7	12
-3	12	18

Table 1: Selection Intervals for S_1

S_2	$64\underline{D}_2$	$64\overline{D}_2$
10	-42	-36
9	-37	-33
8	-33	-29
7	-29	-25
6	-25	-21
5	-22	-18
4	-18	-14
3	-14	-10
2	-10	-6
1	-6	-2
0	-2	3
-1	2	6
-2	6	10
-3	10	14
-4	14	18
-5	18	23
-6	23	27
-7	27	31
-8	31	35
-9	35	39
-10	39	42

Table 2: Selection Intervals for S_2