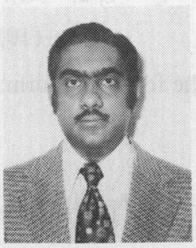
Citroen, Paris, as a Research Engineer working with the control of machine tools. In 1956 he joined the Societe Sciaky, Paris, as Head of the Electronics Laboratory, to design and develop circuits for welding machines. He spent 1959 and 1960 with the Compagnie Generale de Telegraphie sans Fil (CSF), as Head of the Technology Department. From 1961 to 1963 he was a Teaching Assistant at Carnegie-Mellon University. In 1963 joined the University of Alabama, Huntsville, as an Assistant Professor of Electrical Engineering; he became Professor in 1967. He teaches and conducts research in the area of communications and data processing.

Dr. Polge is a member of the Societe Francaise des Electriciens and Sigma Xi.



B.K. Bhagavan was born in Mysore, India, on February 15, 1947. He received the B.E. degree in electrical engineering from the Bangalore University, Bangalore, India, in 1967, the M.E. degree in electrical power engineering from the Indian Institute of Science, Bangalore, in 1969, and the Ph.D. degree from Southern Methodist University, Dallas, Tex., in 1971.

Presently he is a Research Associate with the Research Insitute of the University of Alabama, Hunstsville. His current interests include optimal control, digital processing of image data, and simulation and analysis of radar systems.

James M. Carswell was born in Bishop, Calif., on January 19, 1922.

His major field of interest is mechanical engineering, although he has had considerable experience in electrical engineering. For the last 15 years his work has included trajectory analysis; in particular, digital simulation of six-degree-of freedom trajectories, Monte Carlo analysis of trajectory variables, and special problems in flight mechanics. Until June 1973 he was with the University of Alabama, Huntsville. Currently he is with North American Aviation, Downey, Calif.

Floating-Point Arithmetic Algorithms in the Symmetric Residue Number System

EISUKE KINOSHITA, HIDEO KOSAKO, MEMBER, IEEE, AND YOSHIAKI KOJIMA, SENIOR MEMBER, IEEE

Abstract—The residue number system is an integer number system and is inconvenient to represent numbers with fractional parts. In the symmetric residue system, a new representation of floating-point numbers and arithmetic algorithms for its addition, subtraction, multiplication, and division are proposed. A floating-point number is expressed as an integer multiplied by a product of the moduli. The proposed system assumes existence of necessary conversion procedures before and after the computation.

Index Terms—Cyclic mixed-radix system, exponent part, floating-point arithmetic algorithms, floating-point representation, mantissa, normalized form, number of precision n, symmetric residue number system.

Manuscript received September 10, 1971; revised August 4, 1973.

The authors are with the Department of Electronics, University of Osaka Prefecture, Osaka, Japan.

I. INTRODUCTION

THE residue number system is an integer number system. At present, the techniques known make it inconvenient to represent fractional quantities. It is to be desired that numbers with fractional parts can be handled as easily as integers in the residue number systems.

A few studies on the floating-point arithmetic in the residue system have been published [1], [2]. In these reports a power of 2 or 10 is used as an exponent.

This paper deals with floating-point arithmetic with an exponent which is a product of moduli in the symmetric residue number system. This number system has the following advantages: 1) finding the additive inverse of a residue digit is fairly easy, 2) sign detection by mixed-radix conversion is easy, and 3) the result of scaling is rounded to the closest integer. A cyclic mixed-radix system will be introduced first. Then, based on this system, a new expression of floating-point numbers and algorithms for addition, subtraction, multiplication, and division will be described in terms of normalized operations.

II. Number System

In order to provide the appropriate foundations and motivation for the normalized floating-point format proposed here, a cyclic mixed-radix system will be introduced first.

Cyclic Mixed-Radix System

Consider a weighted number system in which any real number is expressed in the form

$$\pm \sum_{i=-\infty}^{h} \alpha_i w_i \tag{1}$$

where $\{\alpha_i\}$ is a set of permissible digits, $\{w_i\}$ a set of weights, and α_h is the most significant digit of the number. From a practical point of view the series (1) should be approximated by its appropriate partial sum. Here we assume that there are given n radices $m_1, m_2, \cdots, m_n$. We denote by $|i|_n$ the least positive (integer) remainder of the division i/n, and by $[i/n]$ the largest integer less than or equal to i/n, where i is an integer.

Then the approximate value of a given real number may be represented in the form

$$\pm \sum_{i=l}^{l+n-1} \alpha_i w_i \tag{2}$$

with n digits in succeeding positions, where

$$\alpha_i = \alpha_{|i|_n+1} \tag{3}$$

$$0 \leqslant \alpha_{|i|_n+1} \leqslant m_{|i|_n+1} - 1 \tag{4}$$

$$w_i = M^{[i/n]} m_{|i|_n} \cdots m_2 m_1, \qquad (|i|_n \neq 0)$$

$$w_i = M^{[i/n]} \qquad (|i|_n = 0) \tag{5}$$

and α_l is the least significant digit, and where $M = \Pi_{i=1}^{n} m_i$. If a number X is expressed in the form (2), we call X a number of "precision n." It should be noted that numbers in the binary or the decimal system are expressed by the form (2).

Number systems in which the weights are not powers of the same radix are called mixed-radix systems. A cyclic mixed-radix system is defined as a mixed-radix system consisting of all numbers of the form (2). In the following discussion it will be assumed that the radices are odd positive numbers so chosen that $m_1 < m_2 < \cdots < m_n$ and the digits α_i are restricted so that

$$-\frac{m_{|i|_n+1} - 1}{2} \leqslant \alpha_{|i|_n+1} \leqslant \frac{m_{|i|_n+1} - 1}{2} \tag{6}$$

instead of (4). As an example, Table I lists the weights and the permissible digits of the cyclic mixed-radix system with $m_1 = 3, m_2 = 5$, and $m_3 = 7$.

Now consider the numbers of precision n, in a cyclic mixed-radix system, expressed in the form

$$\sum_{i=l}^{l+n-1} \alpha_i w_i \,. \tag{7}$$

Let

$$e = [l/n] \tag{8}$$

$$s = |l|_n \tag{9}$$

$$\sigma_j = |l + j - 1|_n + 1 = |s + j - 1|_n + 1, \qquad (j = 1, 2, \cdots, n). \tag{10}$$

Then, these numbers can be represented in the following form:

$$k \times M^e \times M_s$$

where k is an integer such that $|k| \leqslant \frac{1}{2}(M-1), M = \Pi_{i=1}^{n} m_i$, e is an integer, and

$$M_s = \prod_{i=1}^{s} m_i, \qquad (s \neq 0)$$

$$M_s = 1, \qquad (s = 0).$$

Proof: From (5) and (9),

$$w_l = M^e M_s$$

and moreover, using (10),

$$w_{l+j-1} = m_{\sigma_{j-1}} \cdots m_{\sigma_2} m_{\sigma_1} w_l$$

$$= \prod_{\nu=1}^{j-1} m_{\sigma_\nu} w_l, \qquad (j = 2, 3, \cdots, n).$$

On the other hand, by (3) and (10),

$$\alpha_{l+j-1} = \alpha_{|l+j-1|_n+1} = \alpha_{\sigma_j}, \qquad (j = 1, 2, \cdots, n).$$

Hence

$$\sum_{i=l}^{l+n-1} \alpha_i w_i = \sum_{j=1}^{n} \alpha_{l+j-1} w_{l+j-1}$$

$$= \left\{ \alpha_{\sigma_1} + \sum_{j=2}^{n} \alpha_{\sigma_j} \left(\prod_{\nu=1}^{j-1} m_{\sigma_\nu} \right) \right\} M^e M_s. \tag{11}$$

Here consider the expression (11). Let

$$k = \alpha_{\sigma_1} + \sum_{j=2}^{n} \alpha_{\sigma_j} \left(\prod_{\nu=1}^{j-1} m_{\sigma_\nu} \right). \tag{12}$$

Then, since the α_{σ_j} are integers such that

TABLE I
WEIGHTS AND PERMISSIBLE DIGITS OF CYCLIC MIXED-RADIX SYSTEM WITH $m_1 = 3, m_2 = 5$, AND $m_3 = 7 (n = 3)$

i	[i/n]	$\|i\|_n$	α_i	w_i
⋮	⋮	⋮	⋮	⋮
-5	-2	1	$-2 \leq \alpha_{-5} \leq 2$	$7^{-2} \times 5^{-2} \times 3^{-1}$
-4	-2	2	$-3 \leq \alpha_{-4} \leq 3$	$7^{-2} \times 5^{-1} \times 3^{-1}$
-3	-1	0	$-1 \leq \alpha_{-3} \leq 1$	$7^{-1} \times 5^{-1} \times 3^{-1}$
-2	-1	1	$-2 \leq \alpha_{-2} \leq 2$	$7^{-1} \times 5^{-1}$
-1	-1	2	$-3 \leq \alpha_{-1} \leq 3$	7^{-1}
0	0	0	$-1 \leq \alpha_0 \leq 1$	1
1	0	1	$-2 \leq \alpha_1 \leq 2$	3
2	0	2	$-3 \leq \alpha_2 \leq 3$	5×3
3	1	0	$-1 \leq \alpha_3 \leq 1$	$7 \times 5 \times 3$
4	1	1	$-2 \leq \alpha_4 \leq 2$	$7 \times 5 \times 3^2$
5	1	2	$-3 \leq \alpha_5 \leq 3$	$7 \times 5^2 \times 3^2$
⋮	⋮	⋮	⋮	⋮

$$-\frac{m_{\sigma_j} - 1}{2} \leqslant \alpha_{\sigma_j} \leqslant \frac{m_{\sigma_j} - 1}{2}$$

we obtain

$$|k| \leqslant \left(\frac{m_{\sigma_1}}{2} - \frac{1}{2}\right) + \sum_{j=2}^{n} \left(\frac{m_{\sigma_j}}{2} - \frac{1}{2}\right)\left(\prod_{\nu=1}^{j-1} m_{\sigma_\nu}\right).$$

Therefore,

$$|k| \leqslant \tfrac{1}{2}(m_{\sigma_n} m_{\sigma_{n-1}} \cdots m_{\sigma_1} - 1).$$

In view of (10) it is easily seen that the sequence $\sigma_1, \sigma_2, \cdots, \sigma_n$ is a permutation of $1, 2, \cdots, n$. Hence,

$$|k| \leqslant \tfrac{1}{2}(M - 1).$$

This completes the proof.

It should be noted that (7) can generate negative numbers as well as positive numbers or zero because of the restriction (6).

Normalized Floating-Point Format

In floating-point operation proposed here, the set of numbers consists of 0 and the set of all numbers of the form

$$k \times M^e \times M_s \tag{13}$$

where k is an integer,

$$\tfrac{1}{2}(M/m_s + 1) \leqslant |k| \leqslant \tfrac{1}{2}(M - 1), \quad (s \neq 0)$$

$$\tfrac{1}{2}(M/m_n + 1) \leqslant |k| \leqslant \tfrac{1}{2}(M - 1) \quad (s = 0)$$

normalized by the condition that the most significant digit α_{σ_n} in (12) is not zero, and where e is an integer ranging between $-E$ and E, say. We call the integer k and the pair (e, s) the mantissa and the exponent part, respectively, of the floating number (13). Exceptionally, zero is defined as follows:

$$0 = 0 \times M^0 \times M_0.$$

Theorem: The proposed normalized floating-point format is unique.

Proof: By contradiction.

It suffices to prove the theorem for positive numbers. For a positive number A, let $A = k_1 M^{e_1} M_{s_1}$ and $A = k_2 M^{e_2} M_{s_2}$ with $k_1 \neq k_2$, $e_1 \neq e_2$, or $s_1 \neq s_2$. Assume that the theorem is false; then it must be possible to find an $A = k_1 M^{e_1} M_{s_1} = k_2 M^{e_2} M_{s_2}$ with k_1, e_1, s_1, k_2, e_2, and s_2 meeting the preceding restrictions. Without loss of generality we may assume that $e_2 = e_1 + \Delta e$, where Δe is an integer such that $\Delta e \geqslant 1$. Then it follows that

$$k_1 M_{s_1} = k_2 M^{\Delta e} M_{s_2}$$

which in turn implies

$$M^{\Delta e} = \frac{k_1 M_{s_1}}{k_2 M_{s_2}}$$

where

$$M^{\Delta e} \geqslant M. \tag{14}$$

Since

$$\tfrac{1}{2}(M/m_{s_i} + 1) \leqslant k_i \leqslant \tfrac{1}{2}(M - 1) \tag{15}$$

and

$$1 \leqslant M_{s_i}, \quad (i = 1, 2)$$

then

$$M^{\Delta e} \leqslant \frac{(M-1)M_{s_1}}{\frac{M}{m_{s_2}} + 1}.$$

Further, since

$$\frac{(M-1)M_{s_1}}{\frac{M}{m_{s_2}} + 1} < m_{s_2} M_{s_1} \leqslant M$$

then

$$M^{\Delta e} < M.$$

But this contradicts the assumption (14).
Hence,

$$e_1 = e_2.$$

Next we assume that $s_1 < s_2$ without loss of generality.
From (15),

$$\frac{k_1}{k_2} \leqslant \frac{M - 1}{\frac{M}{m_{s_2}} + 1}.$$

Since

$$\frac{M-1}{\frac{M}{m_{s_2}}+1} < m_{s_2}$$

then

$$\frac{k_1}{k_2} < m_{s_2}. \tag{16}$$

On the other hand, whether $s_1 = 0$ or $s_1 \neq 0$,

$$\frac{k_1}{k_2} = m_{s_1+1} m_{s_1+2} \cdots m_{s_2}$$

but

$$m_{s_1+1} m_{s_1+2} \cdots m_{s_2} \geqslant m_{s_2}$$

hence

$$\frac{k_1}{k_2} \geqslant m_{s_2}.$$

This contradicts (16).
Hence

$$s_1 = s_2.$$

Consequently,

$$k_1 = k_2.$$

This completes the proof of the theorem.

For any real x we denote by fl (x) either of the two numbers of the form (13) which minimize $|\mathrm{fl}(x) - x|$, except that if

$$\tfrac{1}{2} MM^e M_s \leqslant |x| \leqslant \tfrac{1}{4} M^e M_s (2M + m_{s+1} - 1)$$

then

$$\mathrm{fl}(|x|) = \begin{cases} \frac{1}{2}\left(\frac{M}{m_{s+1}} + 1\right) M^e M_{s+1}, & (s \neq n-1) \\ \frac{1}{2}\left(\frac{M}{m_n} + 1\right) M^{e+1} M_0, & (s = n-1). \end{cases}$$

The value fl(x) will be called a normalized floating-point representation of x.

We define the range of floating-point numbers to be the interval

$$D = \left[-\frac{M-1}{2} M^E M_{n-1}, \frac{M-1}{2} M^E M_{n-1}\right].$$

Let fl(x) = $kM^e M_s$ be the normalized floating-point representation of x for $x \in D$, $|x| \geqslant \frac{1}{2}((M/m_n) + 1)M^{-E}$. Then, since

$$\frac{M}{2m_s} M^e M_s \leqslant |x| < \tfrac{1}{2} MM^e M_s, \quad (s \neq 0)$$

$$\frac{M}{2m_n} M^e M_s \leqslant |x| < \tfrac{1}{2} MM^e M_s, \quad (s = 0) \tag{17}$$

it is easily seen that e, s, and k can be determined in the following manner:

$$e = [\log_M 2m_n |x|] - 1.$$

The integer s is found as follows. First compute

$$G_0 = \frac{2|x|}{M^{e+1}}.$$

This quantity is used in the relationship

$$G_i = \frac{G_{i-1}}{m_i}$$

to obtain G_1, G_2, $\cdots$, etc. This iterative procedure is continued until $[G_i] = 0$. If this occurs on the ith ($i = 0, 1, \cdots$), then

$$s = i.$$

Finally k is defined by

$$k = \left(\frac{x}{M^e M_s}\right)_R.$$

Here the R refers to correct rounding in fixed-point arithmetic. If e is outside the interval $-E \leqslant e \leqslant E$, we shall only say that an overflow or an underflow has occurred and shall not proceed further.

Conversion to the Residue Representation

For the integer k in (13) the least remainder in absolute value when divided by m_i may be computed. This quantity, denoted by $/k/_{m_i}$, is referred to as the symmetric residue of k mod m_i, and the radices m_i are called bases or moduli. For any given set of moduli the residues of k may be formed into an n-tuple

$$\{ /k/_{m_1}, /k/_{m_2}, \cdots, /k/_{m_n} \}.$$

This n-tuple is called the symmetric residue representation of k. The integer $/k/_{m_i}$ is called the ith symmetric residue digit

$$-\frac{M-1}{2} \text{ to } \frac{M-1}{2}$$

may be uniquely represented.

Now consider the conversion from a fixed-radix system such as decimal or binary to the residue system. The integer k is specified in a fixed-radix system as

$$k = d_l r^l + d_{l-1} r^{l-1} + \cdots + d_1 r + d_0$$

where r is the radix and $0 \leqslant d_i \leqslant r - 1$. Then, taking this expression modulo m_i, the following equations are obtained:

$$/k/_{m_i} = /d_l /r^l/_{m_i} + d_{l-1} /r^{l-1}/_{m_i} + \cdots + d_1 /r/_{m_i} + d_0/_{m_i}, \quad (i = 1, 2, \cdots, n).$$

Thus, if the powers of r modulo m_i are directly available from the memory, $/k/_{m_i}$ may be computed by repetitive addition (modulo m_i) of those powers of r.

Before going into the main argument, residue interacting operations of mixed-radix conversion, base-extension, and scaling will be briefly described. (For detailed information on these operations, see [3].)

Mixed-Radix Conversion

Mixed-radix conversion process is used to convert from the residue system to the mixed-radix system. The particular mixed-radix representation of interest here is of the form

$$k = \alpha_n m_1 m_2 \cdots m_{n-1} + \cdots + \alpha_2 m_1 + \alpha_1$$

where the α_i are the mixed-radix digits which are to be determined by this procedure. Any integer in the range

$$-\frac{M-1}{2} \text{ to } \frac{M-1}{2}$$

may be represented in this form and hence this representation has the same range as a residue system of moduli $m_1, m_2, \cdots, m_n$.

The mixed-radix conversion is a fundamental operation, from which other important operations such as base extension, relative magnitude comparison, and sign determination can be derived. In the symmetric residue system, the sign of an integer is given by the sign of the most significant nonzero digit of the mixed-radix expression of the number.

By successively subtracting α_i and dividing by m_i in residue notation, all the α_i can be determined, starting with α_1. A simple example of this procedure is given in the Appendix.

Base Extension

Base extension is used to find the residue digits for a new set of moduli, given the residue digits relative to another set of moduli. In most cases, one or more moduli are added to the original base. The procedure is a mixed-radix conversion with an additional final step. A simple example of base extension is given in the Appendix.

Scaling

In conventional fixed-radix arithmetic, scaling up or down by a power of the radix is simply a series of right or left shifts and is a fast economical operation. In the residue number system, because multiplication is a simple operation, scaling up is no problem. Scaling down is, in general, a difficult operation. However, it is easy to scale down by a product of the m_i; for example, permissible divisors are $m_1 \times m_2 \times m_5$ or m_2 but not $m_1 \times m_2^2 \times m_5$ or m_2^2. This restricted operation is referred to as scaling. A simple example of scaling is given in the Appendix.

III. Floating-Point Arithmetic

Let $X = k_x M^{e_x} M_{s_x}$ and $Y = k_y M^{e_y} M_{s_y}$ be two floating-point numbers in the range D, where the subscripts x and y represent X and Y, respectively. We define by $Z = \mathrm{fl}(X * Y)$ the desired result of a floating-point operation, where Z is of normalized form and the $*$ symbol represents addition, subtraction, multiplication, or division of two floating-point numbers.

Adjustment and Overflow or Underflow in Mantissa Part

In the proposed floating-point number system, some adjustment of a mantissa part is needed before computation in multiplication or division as well as in addition or subtraction.

Overflow or underflow occurs any time the mantissa of $X * Y$ would fall, in absolute value, outside the interval

$$I = \begin{cases} \left[\frac{1}{2}\left(\frac{M}{m_s}+1\right), \frac{1}{2}(M-1)\right], & (s \neq 0) \\ \left[\frac{1}{2}\left(\frac{M}{m_n}+1\right), \frac{1}{2}(m-1)\right], & (s = 0). \end{cases}$$

Then, special methods are needed to detect the occurrence of overflow or underflow and to normalize the result.

Addition or Subtraction

We assume, without loss of generality, that

$$M^{e_x} M_{s_x} \geqslant M^{e_y} M_{s_y}. \tag{18}$$

The exponent parts of X and Y must be made equal before addition or subtraction. To align the exponent parts, we rewrite Y as

$$Y = K M^{e_x} M_{s_x}$$

where

$$K = \frac{k_y M^{e_y - e_x} M_{s_y}}{M_{s_x}}.$$

Then, we can express the sum or difference of X and Y as

$$k_z M^{e_z} M_{s_z}$$

where

$$k_z = k_x \pm K_R, \quad e_z = e_x, \quad s_z = s_x.$$

It should be noted that, in general, the floating-point number is not in the normalized form at this point.

Now in order to know how to get K_R, consider the relationship between e_x and e_y, or s_x and s_y.

1) If $e_x \geqslant e_y + 2$,

$$|K| \leqslant |k_y| M^{-2} M_{s_y} / M_{s_x} \leqslant |k_y| / (M m_n) < 1/(2m_n).$$

Hence

$$K_R = 0.$$

2) If $e_x = e_y + 1$ and $s_x \geqslant s_y$,

$$|K| = |k_y| M^{-1} M_{s_y} / M_{s_x} \leqslant |k_y| / M < 1/2.$$

Hence

$$K_R = 0.$$

3) If $e_x = e_y + 1$ and $s_x < s_y$,

$$K = k_y M^{-1} M_{s_y} / M_{s_x} = \frac{k_y m_1 m_2 \cdots m_{s_y}}{m_0 m_1 \cdots m_{s_x} m_1 m_2 \cdots m_n}$$

$$= \frac{k_y}{m_0 m_1 \cdots m_{s_x} m_{s_y+1} m_{s_y+2} \cdots m_n}$$

where $m_0 = 1$. In the symmetric residue system the result of scaling is rounded to the closest integer, hence K_R is obtained by scaling k_y by $m_0 m_1 \cdots m_{s_x} m_{s_y+1} m_{s_y+2} \cdots m_n$. Here, by the assumption that all the moduli are odd, $|K_R - K| < 1/2$.

4) If $e_x = e_y$ and $s_x > s_y$,

$$K = k_y M_{s_y} / M_{s_x} = \frac{k_y}{m_{s_y+1} m_{s_y+2} \cdots m_{s_x}}.$$

Hence K_R is obtained by scaling k_y by $m_{s_y+1} m_{s_y+2} \cdots m_{s_x}$.

5) If $e_x = e_y$ and $s_x = s_y$, then evidently

$$K_R = k_y.$$

Note that the only remaining case $e_x = e_y$ and $s_x < s_y$ never occurs by virtue of the assumption (18).

After making necessary arrangement for an alignment of exponent parts as is previously stated, we compute $k_z = k_x \pm K_R$. Then, since $0 \leqslant |k_z| \leqslant M - 1$, an overflow or an underflow may have occurred, and a test is required for overflow or underflow detection.

Consider the symmetric residue system with moduli m_1, $m_2, \cdots, m_n$ and a redundant odd modulus m_{n+1}, where m_{n+1} is pairwise relatively prime to all the other moduli and satisfies the conditions

$$M - 1 \leqslant \tfrac{1}{2}(Mm_{n+1} - 1), \quad m_n < m_{n+1}. \tag{19}$$

Suppose we have the residue representations of k_x and k_y for all moduli, including m_{n+1}. If k_z is expressed in its mixed-radix form, we have

$$k_z = \alpha_{n+1} m_1 m_2 \cdots m_n + \alpha_n m_1 m_2 \cdots m_{n-1} + \cdots + \alpha_2 m_1 + \alpha_1.$$

Define α_u to be the most significant mixed-radix digit which is not equal to zero. The subscript u will be equal to some integer from $n + 1$ through 1. Then, u is equal to n if and only if $\frac{1}{2}((M/m_n) + 1) \leqslant |k_z| \leqslant \frac{1}{2}(M - 1)$. Hence an overflow has occurred if $u = n + 1$. If $u < n$, then

$$|k_z| < \tfrac{1}{2}\left(\frac{M}{m_n} + 1\right) \leqslant \tfrac{1}{2}\left(\frac{M}{m_{s_z}} + 1\right)$$

which implies that an underflow has occurred. The only remaining case is if $u = n$. In this case it is not possible to know from u alone if an underflow has occurred, and another test is required.

A method for the underflow detection requires the availability of the quantities $\frac{1}{2}((M/m_{s_z}) + 1)(s_z = 1, 2, \cdots, n - 1)$ modulo $m_i (i = 1, 2, \cdots, n)$. If $s_z = 0$, $\frac{1}{2}((M/m_{s_z}) + 1) \leqslant |k_z|$, which implies that no underflow has occurred. The quantities $\frac{1}{2}((M/m_{s_z}) + 1)$ are constants and can be permanently stored in the residue form. If $s_z \neq 0$, $|k_z| - \frac{1}{2}((M/m_{s_z}) + 1)$ is formed in its residue code and converted to its mixed-radix form. If the sign of the most significant nonzero digit of this form is negative, an underflow has occurred.

After these tests, k_z is replaced, if necessary, by the integer obtained by scaling or multiplying k_z by a factor, and the exponent part (e_z, s_z) is corrected.

In scaling k_z, it is desirable to choose m_{s_z+1} as the factor, taking account of the definition of M_{s_z}, and increase s_z by 1. If s_z has reached n, s_z is set to zero and e_z is increased by 1. It can be shown easily that $|k_z|$ falls into the interval I after being scaled by m_{s_z+1}.

In multiplying k_z by a factor, if $u < n$, since $k_z M^{e_z} M_{s_z}$ can be expressed as

$$k_z \left(\prod_{i=s_z+u+1}^{s_z+n} m_i \right) M^{e_z - 1} \prod_{j=1}^{s_z+u} m_j$$

we choose $\Pi_{i=s_z+u+1}^{s_z+n} m_i$ as the factor, decrease e_z by 1 and increase s_z by u, where i is a modulo n number when $i > n$. If s_z has reached n, s_z is set to zero and e_z is increased by 1. The quantities $\Pi_{i=s_z+u+1}^{s_z+n} m_i$ are constants determined by s_z and u and can be permanently stored in the residue form. If $u = n$, we choose m_{s_z} as the factor and decrease s_z by 1. The quantities $m_{s_z}(s_z = 1, 2, \cdots, n - 1)$ are constants and can be permanently stored in the residue form. It can be shown that k_z is normalized after a finite number of this multiplicative iterations. In the case of $u = 1$ or $u = n$ only one iteration is required.

Thus we have the desired result $\mathrm{fl}(X \pm Y) = k_z M^{e_z} M_{s_z}$ which is the floating-point representation of $X \pm Y$.

Multiplication

It will be assumed, without loss of generality, that $s_x \geqslant s_y$. We first consider the case of $s_y = 0$ or 1. In this case, the product of X and Y can be expressed in the form

$$k_z M^{e_z} M_{s_z}$$

where

$$k_z = k_x k_y M_{s_y}, \quad e_z = e_x + e_y, \quad s_z = s_x.$$

Obviously this is an unnormalized number. If $s_y = 0$,

$$\tfrac{1}{4}\left(\frac{M}{m_n} + 1\right)^2 \leqslant |k_z| \leqslant \tfrac{1}{4}(M - 1)^2$$

and if $s_y = 1$,

$$\frac{1}{4}\left(\frac{M}{m_1}+1\right)\left(\frac{M}{m_{n-1}}+1\right)m_1 \leqslant |k_z| \leqslant \frac{1}{4}(M-1)^2 m_1.$$

If we suppose that

$$\frac{1}{2}(M+1) \leqslant \frac{1}{4}\left(\frac{M}{m_n}+1\right)^2$$

then multiplicative overflow occurs always in computing k_z. To cope with this difficulty, consider the symmetric residue system with redundant odd moduli $m_{n+2}, m_{n+3}, \cdots, m_{n+p}$ in addition to $m_1, m_2, \cdots, m_{n+1}$, where $m_i (i = n+2, n+3, \cdots, n+p)$ are pairwise relatively prime to all the other moduli and satisfy the conditions

$$p \leqslant n$$

$$\frac{1}{4}(M-1)^2 m_1 \leqslant \frac{1}{2}(M m_{n+1} m_{n+2} \cdots m_{n+p} - 1),$$

$$m_{n+1} < m_{n+2} < \cdots < m_{n+p-1}. \tag{20}$$

Before computation of k_z, we extend the base to include $m_{n+2}, m_{n+3}, \cdots, m_{n+p}$ for the residue representations of k_x and k_y. The multiplier M_1 required for getting k_z when $s_y = 1$ is constant and can be permanently stored in the residue form.

Now to normalize k_z, define α_u to be the most significant nonzero mixed-radix digit of k_z. Then, if u is greater than n, overflow has occurred, and therefore k_z must be replaced by the result of scaling k_z by a factor. The scaling factor is chosen as follows.

For u such that $u \geqslant n+2$, we choose $\Pi_{i=s_z+1}^{s_z+u-n} m_i$ as factor and increase s_z by $u-n$, where i is a modulo n number when $i > n$. If s_z has reached n, s_z is set to zero and e_z is increased by 1. It can be shown that the value of u decreases to $u = n+1$ after a finite number of this scaling iterations when $u \geqslant n+2$.

If u has reached $n+1$, k_z is scaled by m_{s_z+1} and s_z is increased by 1. If s_z has reached n, s_z is set to zero and e_z is increased by 1. It can be easily shown that

$$\frac{1}{2}\left(\frac{M}{m_{s_z+1}}+1\right) \leqslant \left(\frac{|k_z|}{m_{s_z+1}}\right)_R \leqslant \frac{1}{2}\left(\frac{M m_{n+1}}{m_{s_z+1}}-1\right)$$

and

$$\frac{1}{2}(M-1) < \frac{1}{2}\left(\frac{M m_{n+1}}{m_{s_z+1}}-1\right)$$

which implies an overflow may have occurred.

At this point, to detect an overflow $|k_z| - \frac{1}{2}(M-1)$ is formed in its residue code and converted to its mixed-radix form. If the sign of the most significant nonzero digit of this form is positive, an overflow has occurred, then k_z is scaled by m_{s_z+1} and the exponent part is corrected. The quantity $\frac{1}{2}(M-1)$ can be permanently stored in the residue form modulo $m_i (i = 1, 2, \cdots, n+1)$.

Thus we have the desired result $\mathrm{fl}(X \times Y) = k_z M^{e_z} M_{s_z}$ in the case of $s_y = 0$ or 1. A similar procedure is applicable to the case of $s_y \neq 0$ and $s_y \neq 1$, using the following modifications. In this case we can express the product of X and Y as

$$k_z M^{e_z} M_{s_z}$$

where

$$k_z = \left(\frac{k_x k_y}{m_{s_y+1} m_{s_y+2} \cdots m_n}\right)_R, \quad e_z = e_x + e_y + 1, \quad s_z = s_x.$$

First of all, we extend the base to include $m_{n+2}, m_{n+3}, \cdots, m_{n+p}$ for the residue representations of k_x and k_y, and then compute $k_x k_y$. Scaling $k_x k_y$ by $m_{s_y+1} m_{s_y+2} \cdots m_n$, we have k_z. It can be easily shown that $\frac{1}{2}((M/m_{s_z}) + 1) < |k_z|$. Hence k_z must be normalized by the aforementioned scaling algorithm.

Division

Consider the division of X by Y. If $Y = 0$, division is not defined. Otherwise, since $|k_x|, |k_y| \leqslant \frac{1}{2}(M-1)$, it is impossible to get the desired precision of the quotient k_x/k_y by merely dividing k_x by k_y.

To increase the precision, we define the following function A which is a multiplier to k_x.

1) If $s_x \geqslant s_y$ and $s_y = 0$,

$$\frac{X}{Y} = \frac{k_x M}{k_y} M^{e_x - e_y - 1} M_{s_x}. \tag{21}$$

Hence A is defined by

$$A = M.$$

2) If $s_x \geqslant s_y$ and $s_y \neq 0$,

$$\begin{aligned}\frac{X}{Y} &= \frac{k_x B C M_{s_x}}{k_y M_{s_y} B C} M^{e_x - e_y} \\ &= \frac{k_x B C M_{s_x}}{k_y M C} M^{e_x - e_y} \\ &= \frac{k_x B C}{k_y} M^{e_x - e_y - 1} M_{s_x - s_y}\end{aligned} \tag{22}$$

where

$$B = \prod_{i=s_y+1}^{n} m_i, \quad C = \prod_{i=s_x-s_y+1}^{s_x} m_i.$$

Hence

$$A = B \times C.$$

Note that if $s_x = s_y$ and $s_y \neq 0$, then $A = M$.

3) If $s_x < s_y$,

$$\frac{X}{Y} = \frac{k_x M_{s_x} B}{k_y M_{s_y} B} M^{e_x - e_y}$$

$$= \frac{k_x M_{s_x} B}{k_y M} M^{e_x - e_y}$$

$$= \frac{k_x M_{s_x} B}{k_y} M^{e_x - e_y - 1}$$

$$= \frac{k_x M_{s_x} BD}{k_y} M^{e_x - e_y - 2} M_{n+s_x-s_y} \quad (23)$$

where

$$D = \prod_{i=n+s_x-s_y+1}^{n} m_i.$$

Hence A is defined by

$$A = M_{s_x} \times B \times D.$$

The quantity A is a function of s_x and s_y and is a product of n moduli. It can be easily seen that A has a maximum

$$A_{\max} = \begin{cases} (m_n m_{n-1} \cdots m_{(n+3)/2})^2 m_{(n+1)/2}, & (n \text{ is odd}) \\ (m_n m_{n-1} \cdots m_{(n/2)+1})^2, & (n \text{ is even}) \end{cases}$$

when $s_x = 0$ and $s_y = [n/2]$.

It follows that $|k_x| A \leqslant \frac{1}{2}(M-1)A_{\max}$. Therefore, the previous redundant moduli $m_{n+2}, m_{n+3}, \cdots, m_{n+p}$ are chosen so that they further satisfy the following condition:

$$\tfrac{1}{2}(M-1)A_{\max} \leqslant \tfrac{1}{2}(Mm_{n+1}m_{n+2} \cdots m_{n+p} - 1). \quad (24)$$

Before the computation, we extend the base to include $m_{n+2}, m_{n+3}, \cdots, m_{n+p}$ for the residue representations of k_x and k_y. Then we form the product of k_x and A and divide $k_x A$ by k_y. The value of A is a constant determined by s_x and s_y and can be permanently stored in the residue form. It is to be desired that the integer quotient $k_z = (k_x A/k_y)_R$ should be found by applying the division algorithm proposed by the authors [4].

The equations required for calculation of exponent part is given as follows.

If $s_x \geqslant s_y$, from (21) and (22),

$$e_z = e_x - e_y - 1, \quad s_z = s_x - s_y$$

and if $s_x < s_y$ from (23),

$$e_z = e_x - e_y - 2, \quad s_z = n + s_x - s_y.$$

Thus we have an unnormalized floating number

$$k_z M^{e_z} M_{s_z}.$$

It can be shown that no underflow has occurred and an overflow may have occurred in computing $k_x A/k_y$. If an overflow has occurred, the same discussion and procedure as described in the case of multiplication may be applied to normalization of k_z.

IV. Sample Residue Number System

As a sample residue number system, consider the symmetric residue system consisting of moduli $m_1 = 13, m_2 = 17, m_3 = 19, m_4 = 23$, and $m_5 = 29$ ($M = 2800733$). Then, for the mantissa of the floating-point number (13), the interval of definition is, in absolute value,

$$\begin{array}{ll} [\ 48289, 1400366] & (s = 0) \\ [107721, 1400366] & (s = 1) \\ [\ 82375, 1400366] & (s = 2) \\ [\ 73704, 1400366] & (s = 3) \\ [\ 60886, 1400366] & (s = 4). \end{array}$$

The redundant moduli which satisfy (19), (20), and (24) are chosen as $m_6 = 31, m_7 = 37, m_8 = 41, m_9 = 43$, and $m_{10} = 11$.

In floating-point add operation, underflow detection requires a table of the quantities $L_{s_z} = \frac{1}{2}((M/m_{s_z}) + 1)(s_z = 1, 2, 3, 4)$. This can be accomplished by storing, in a special memory, a table such as Table II. The symbol $((\ ))_5$ stands for the residue representation, for instance, $((L_1))_5 \longleftrightarrow \{|L_1|_{m_1}, |L_1|_{m_2}, \cdots, |L_1|_{m_5}\}$. Normalization requires tables of $\prod_{i=s_z+u+1}^{s_z+n} m_i$ (for $u < n$) and m_{s_z} (for $u = n$) such as Tables III and IV.

For floating-point multiplication, the quantity M_1 must be provided to get $k_z = k_x k_y M_{s_y}$ when $s_y = 1$. This can be accomplished by storing $((m_1))_{10}$ in a special memory. Overflow detection requires the residue representation of $((\frac{1}{2}(M-1)))_6$.

To have the desired precision of a quotient in floating-point division, we must provide a table of multipliers A, such as Table V.

V. Algorithms

From Section III, we may summarize the algorithms for floating-point residue arithmetic as follows.

Suppose k_x and k_y to be represented by symmetric residue digits $((k_x))_{n+1}$ and $((k_y))_{n+1}$.

Floating-Point Add or Subtract Operation

Assume that $X = k_x M^{e_x} M_{s_x}$ and $Y = k_y M^{e_y} M_{s_y}$ are the augend (minuend) and the addend (subtrahend), respectively, and that $M^{e_x} M_{s_x} \geqslant M^{e_y} M_{s_y}$.

1) Compute $\Delta e = e_x - e_y$. If $\Delta e \geqslant 2$, fl$(X \pm Y) = X$.

2) If $\Delta e = 1$, check whether $s_x \geqslant s_y$. If this is true fl $(X \pm Y) = X$. Otherwise, scale k_y by $m_0 m_1 \cdots m_{s_x} m_{s_y+1} m_{s_y+2} \cdots m_n$ and let the result to be K_R.

TABLE II
TABLE COMPOSED OF $L_{s_z} = \frac{1}{2}((M/m_{s_z}) + 1)$

s_z	L_{s_z}
1	$((L_1))_5$
2	$((L_2))_5$
3	$((L_3))_5$
4	$((L_4))_5$

TABLE III
TABLE COMPOSED OF $\Pi_{i=s_z+u+1}^{s_z+n} m_i$

u \ s_z	0	1	2	3	4
1	$((m_2m_3m_4m_5))_6$	$((m_3m_4m_5m_1))_6$	$((m_4m_5m_1m_2))_6$	$((m_5m_1m_2m_3))_6$	$((m_1m_2m_3m_4))_6$
2	$((m_3m_4m_5))_6$	$((m_4m_5m_1))_6$	$((m_5m_1m_2))_6$	$((m_1m_2m_3))_6$	$((m_2m_3m_4))_6$
3	$((m_4m_5))_6$	$((m_5m_1))_6$	$((m_1m_2))_6$	$((m_2m_3))_6$	$((m_3m_4))_6$
4	$((m_5))_6$	$((m_1))_6$	$((m_2))_6$	$((m_3))_6$	$((m_4))_6$

TABLE IV
TABLE COMPOSED OF m_{s_z}

s_z	m_{s_z}
1	$((m_1))_6$
2	$((m_2))_6$
3	$((m_3))_6$
4	$((m_4))_6$

TABLE V
TABLE COMPOSED OF A VALUES

s_y \ s_x	0	1	2	3	4
0	$((M))_{10}$	$((M))_{10}$	$((M))_{10}$	$((M))_{10}$	$((M))_{10}$
1	$((m_2m_3m_4m_5^2))_{10}$	$((M))_{10}$	$((m_2^2m_3m_4m_5))_{10}$	$((m_2m_3^2m_4m_5))_{10}$	$((m_2m_3m_4^2m_5))_{10}$
2	$((m_3m_4^2m_5^2))_{10}$	$((m_1m_3m_4m_5^2))_{10}$	$((M))_{10}$	$((m_2m_3^2m_4m_5))_{10}$	$((m_3^2m_4^2m_5))_{10}$
3	$((m_3m_4^2m_5^2))_{10}$	$((m_1m_4^2m_5^2))_{10}$	$((m_1m_2m_4m_5^2))_{10}$	$((M))_{10}$	$((m_2m_3m_4^2m_5))_{10}$
4	$((m_2m_3m_4m_5^2))_{10}$	$((m_1m_3m_4m_5^2))_{10}$	$((m_1m_2m_4m_5^2))_{10}$	$((m_1m_2m_3m_5^2))_{10}$	$((M))_{10}$

3) If $\Delta e = 0$, check whether $s_x = s_y$. If so, set $K_R = k_y$. Otherwise (if $s_x > s_y$), scale k_y by $m_{s_y+1}m_{s_y+2} \cdots m_{s_x}$ and let the result to be K_R.

4) Compute $k_z = k_x \pm K_R$ and set $e_z = e_x$, $s_z = s_x$.

5) Find the mixed-radix digits $\alpha_1, \alpha_2, \cdots, \alpha_{n+1}$ of k_z which are associated with the symmetric residue system $m_1, m_2, \cdots, m_{n+1}$. If all the digits $\alpha_i (i = 1, 2, \cdots, n + 1)$ are zero, set $e_z = 0, s_z = 0$.

6) Otherwise, denote by α_u the most significant nonzero digit. If $u = n + 1$, *scale* k_z by m_{s_z+1} and increase s_z by 1. If s_z has reached n, set $s_z = 0$ and increase e_z by 1.

7) If $u < n$, multiply k_y by $\Pi_{i=s_z+u+1}^{s_z+n} m_i$ which can be

read from the table by means of u and s_z, decrease e_z by 1, and increase s_z by u. Then, if $s_z < n$, return to Step 5). If $s_z \geqslant n$, after increasing e_z by 1 and decreasing s_z by n, return to Step 5).

8) If $u = n$, check whether $s_z = 0$. If so, k_z has been already normalized. If $s_y \neq 0$, form $|k_z| - \frac{1}{2}((M/m_{s_z}) + 1)$ and find the sign of the result through the mixed-radix conversion, where $\frac{1}{2}((M/m_{s_z}) + 1)$ is read from a memory. If the sign is negative, multiply k_z by m_{s_z} and decrease s_z by 1.

Thus we can get the desired result fl $(X \pm Y) = k_z M^{e_z} M_{s_z}$.

Example 1: In the same residue system in Section IV, suppose we add $X = 1398046M^1 M_4$ and $Y = 1258463M^1 M_2$. Examining these two numbers we find $e_x - e_y = 0$ and $s_x > s_y$. The mantissa of Y must be scaled by $m_3 m_4$. This gives

$$K_R = \left(\frac{1258463}{19 \times 23}\right)_R = 2880.$$

Adding k_x and K_R, we get $k_z = 1400926$, then we set $e_z = 1$, $s_z = 4$. Replacing k_z by $(k_z/m_5)_R$ and correcting the exponent part, we get

$$k_z = 48308, \quad e_z = 2, \quad s_z = 0.$$

The desired result then is

$$\text{fl}\,(1398046M^1 M_4 + 1258463M^1 M_2) = 48308M^2 M_0.$$

Floating-Point Multiply Operation

Assume that $X = k_x M^{e_x} M_{s_x}$ and $Y = k_y M^{e_y} M_{s_y}$ are the multiplicand and the multiplier, respectively, and that $s_x \geqslant s_y$.

1) Perform the base-extension operation on k_x and k_y and find the symmetric residue digits $/k_x/_{m_i}$ and $/k_y/_{m_i}$ $(i = n + 2, n + 3, \cdots, n + p)$.

2) Set $e_z = e_x + e_y$, $s_z = s_x$.

3) If $s_y = 0$, compute $k_z = k_x k_y$, and if $s_y = 1$, compute $k_z = k_x k_y m_1$, where m_1 is fetched out from a memory. Otherwise, increase e_z by 1, compute the product $k_x k_y$ and scale this result by $m_{s_y+1} m_{s_y+2} \cdots m_n$ to get $k_z = (k_x k_y/(m_{s_y+1} m_{s_y+2} \cdots m_n))_R$.

4) Find the mixed-radix digits $\alpha_1, \alpha_2, \cdots, \alpha_{n+p}$ of k_z. If all the digits α_i are zero, set $e_z = 0$, $s_z = 0$. Otherwise denote by α_u the most significant nonzero digit.

5) If $u \geqslant n + 2$, replace k_z by $(k_z/\Pi_{i=s_z+1}^{s_z+u-n} m_i)_R$ and increase s_z by $u - n$, where i is a modulo n number when $i > n$. If $s_z = n$, set s_z to zero and increase e_z by 1. Return to Step 4).

6) If $u = n + 1$, replace k_z by $(k_z/m_{s_z+1})_R$ and increase s_z by 1. If $s_z = n$, set s_z to zero and increase e_z by 1. Compute $|k_z| - \frac{1}{2}(M - 1)$ and find the sign of this result by means of the mixed-radix conversion, where $\frac{1}{2}(M - 1)$ is fetched out from a memory. If the sign is positive, replace k_z by $(k_z/m_{s_z+1})_R$ and increase s_z by 1. If $s_z = n$, set s_z to zero and increase e_z by 1.

Thus we can get the desired result fl $(X \times Y) = k_z M^{e_z} M_{s_z}$.

Example 2: In the preceding residue system, suppose we multiply $X = 310418M^0 M_4 = \text{fl}\,(2.99792 \times 10^{10})$ by $Y = 1141783M^{-5} M_0 = \text{fl}\,(6.6256 \times 10^{-27})$.

First, we set $e_z = 0 + (-5) = -5$, $s_z = 4$. Since $s_y = 0$,

$$k_z = k_x k_y = 354429995294.$$

We can easily know that the value of u is 10. Hence, scaling k_z by M, we have $k_z = 126549$, $e_z = -4$, $s_z = 4$. This mantissa is in the normalized form. Thus we have

$$\text{fl}\,(310418M^0 M_4 \times 1141783M^{-5} M_0) = 126549M^{-4} M_4.$$

Floating-Point Divide Operation

Assume that $X = k_x M^{e_x} M_{s_x}$ and $Y = k_y M^{e_y} M_{s_y}$ are the dividend and divisor, respectively.

1) Perform the base extension on k_x and k_y and find the symmetric residue digits $/k_x/_{m_i}$ and $/k_y/_{m_i}$ $(i = n + 2, n + 3, \cdots, n + p)$.

2) Multiply k_x by A, which can be read from a memory by means of s_x and s_y, and set $e_z = e_x - e_y - 1$, $s_z = s_x - s_y$.

3) If $s_x < s_y$, decrease e_z by 1 and increase s_z by n.

4) Compute $k_z = (k_x A/k_y)_R$ (check if $k_y = 0$; if so, division is not defined).

Hereafter, follow from Step 4) to Step 6) in the multiply algorithm.

Example 3: Let

$$X = 126549M^{-4} M_4$$

$$Y = 87955M^{-4} M_4 = \text{fl}\,(1.38054 \times 10^{-16}).$$

Then, since $s_x = s_y = 4$, $A = M$.

$$e_z = -4 - (-4) - 1 = -1, s_z = 0.$$

$$k_z = \left(\frac{k_x A}{k_y}\right)_R = 4029674.$$

The subscript u for 4029674 is 6. Hence, scaling k_z by m_1, we have $k_z = 309975$, $e_z = -1$, $s_z = 1$. Thus we have

$$\text{fl}\,(126549M^{-4} M_4/(87955M^{-4} M_4)) = 309975M^{-1} M_1.$$

VI. Conclusion

A new residue floating-point number expression and arithmetic algorithms based on it have been proposed.

The main advantage of the residue system is that in addition, subtraction, and multiplication any particular digit of the result is dependent only on the corresponding operand digits. This property eliminates carries from digit to digit for all three arithmetic operations previously mentioned and removes the need for partial product formation in multiplication. In contrast to conventional digital systems, these three operations can be executed in the same time as required for an addition operation, while the following operations have been criticized as awkward in residue computers: detecting sign and the occurrence of overflow, relative-magnitude comparison of two numbers, extending the range of the number system, and shifting the operand.

For residue arithmetic, the use of matrix units is most attractive for implementing addition, subtraction, and multiplication modulo m_i. This method of implementation has the advantage of the absence of carries inside as well as among residue digits. However, a disadvantage arises when mechanizing the matrices. In general, 1-out-of-m_i coding is used to drive the matrix; therefore, the number of components required for each matrix is large. The use of the symmetric residue notation permits folding the arithmetic matrices, which results in a decrease in matrix component. A desirable form of logical implementation would be the use of the arithmetic matrices realized with LSI techniques. Such matrix units would give extremely high speed to the three residue operations.

The proposed expression is basically different from the other residue floating-point number expression. It represents a residue floating-point number as an integer multiplied by a product of the moduli. As a result the newly proposed expression has the advantage of efficient application of residue interacting operations to the floating-point arithmetics. These interacting operations are split into base extension, scaling, and mixed-radix conversion and are closely connected with conventional residue arithmetic operations. There is much possibility of this advantage being regarded conversely as disadvantage because these interacting operations are usually taken as time consuming. The proposed algorithms have the obvious advantage of entire performance of multiplication and division as well as addition and subtraction, under a certain condition of the moduli used.

The algorithms make good use of the interacting operations which can cope with the awkward operations mentioned previously. The main disadvantage of the residue interacting operations is that these operations are relatively time consuming because of the cascaded process used. This disadvantage, however, would be little worth consideration when the aforementioned LSI logic matrices are used, because the residue interacting operations consist of residue addition, subtraction, and multiplication.

Conversion before or after the computation will be discussed in another paper under preparation.

The material of this paper forms an investigation of the applicability of residue number system to the floating-point arithmetics.

Appendix

A. Example of Mixed-Radix Conversion

For $m_1 = 7$, $m_2 = 11$, $m_3 = 13$, and $m_4 = 17$, find the symmetric mixed-radix digits of $X \longleftrightarrow \{2, -5, 2, -2\}$.

Solution

Moduli	7	11	13	17	
Residue representation of X	2	−5	2	−2	$\alpha_1 = 2$
Subtract α_1		2	2	2	
		4	0	−4	
Multiply by $/\frac{1}{7}/_{m_i}$		−3	2	5	
		−1	0	−3	$\alpha_2 = -1$
Subtract α_2			−1	−1	
			1	−2	
Multiply by $/\frac{1}{11}/_{m_i}$			6	−3	
			6	6	$\alpha_3 = 6$
Subtract α_3				6	
				0	

At this point, it should be apparent that the remaining mixed-radix digit α_4 must be zero. Hence the process can be terminated. Thus we have

$$X = 6(7 \times 11) + (-1)(7) + 2(1) = 457.$$

In the preceding, the quantity of the form $/\frac{1}{a}/_{m_i}$ is called the multiplicative inverse of a mod m_i. The quantity $/\frac{1}{a}/_{m_i}$ exists if and only if $(a, m_i) = 1$ and $/a/_{m_i} \neq 0$ and satisfies

$$-\frac{m_i - 1}{2} \leqslant /\tfrac{1}{a}/_{m_i} \leqslant \frac{m_i - 1}{2} \quad \text{and} \quad \Big/ /\tfrac{1}{a}/_{m_i} \times a \Big/_{m_i} = 1.$$

B. Example of Base-Extension

Given the residue representation $X = 37 \longleftrightarrow \{2, 4\}$ for the base with moduli 7 and 11, find $/X/_{13}$ and $/X/_{17}$.

Solution

Moduli	7	11	13	17	
Residue representation of X	2	4	0	0	$\alpha_1 = 2$
Subtract α_1		2	2	2	
		2	−2	−2	
Multiply by $/\frac{1}{7}/_{m_i}$		−3	2	5	
		5	−4	7	$\alpha_2 = 5$
Subtract α_2			5	5	
			4	2	

Then

$$/\tfrac{1}{7} \times /X/_{13} + 4/_{13} = 0$$

$$/\tfrac{1}{7} \times /X/_{17} + 2/_{17} = 0.$$

Hence

$$/\tfrac{1}{7} \times /X/_{13}/_{13} = -4,$$

$$/\tfrac{1}{7} \times /X/_{17}/_{17} = -2.$$

Consequently,

$$/X/_{13} = -2$$

$$/X/_{17} = 3.$$

C. Example of Scaling

For $m_1 = 7, m_2 = 11, m_3 = 13$, and $m_4 = 17$, scale $X = 6826 \longleftrightarrow \{1, -5, 1, -8\}$ by the scale factor 11×17. Denote the result by Z.

Solution

Moduli	7	11	13	17
Residue representation of X	1	-5	1	-8
Subtract $/X/_{11} = -5$	2		-5	-5
	-1		6	3
Multiply by $/\frac{1}{11}/_{m_i}$	2		6	-3
	-2		-3	-8
Subtract $\left/\dfrac{X-/X/_{11}}{11}\right/_{17} = -8$	1		5	
	-1		5	
Multiply by $/\frac{1}{17}/_{m_i}$	-2		-3	
Enter 0 into missing columns for extension of base	2	0	-2	0
Subtract 2		2	2	2
		-2	-4	-2
Multiply by $/\frac{1}{7}/_{m_i}$		-3	2	5
		-5	5	7
Subtract 5		5		5
		1		2

Then $/\frac{1}{7} \times /Z/_{11} + 1/_{11} = 0$ and $/\frac{1}{7} \times /Z/_{17} + 2/_{17} = 0$. Hence $/Z/_{11} = 4$ and $/Z/_{17} = 3$. Therefore, the residue representation of $6873/(11 \times 17) \cong 37$ is $\{2, 4, -2, 3\}$. Note that $6826/(11 \times 17) \approx 36.503$ and hence it was rounded to 37, the closest integer, rather than to 36.

Acknowledgment

The authors would like to thank the referees for their encouragement and very helpful suggestions.

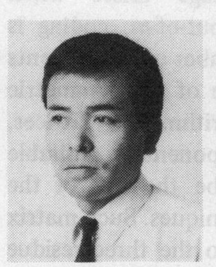
References

[1] A. Svoboda, *Digitale Informationswandler.* Braunschweig, Germany: Vieweg and Sohn, 1960.

[2] A. Sasaki, "The basis for implementation of additive operations in the residue number system," *IEEE Trans. Comput.*, vol. C-17, pp. 1066-1073, Nov. 1968.

[3] N.S. Szabo and R.I. Tanaka, *Residue Arithmetic and its Applications to Computer Technology.* New York: McGraw-Hill, 1967.

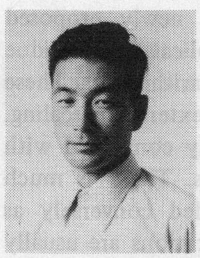
[4] E. Kinoshita, H. Kosako, and Y. Kojima, "General division in the symmetric residue number system," *IEEE Trans. Comput.*, vol. C-22, pp. 134-142, Feb. 1973.



Eisuke Kinoshita was born in Osaka, Japan, in 1932. He received the B.S. degree in mathematics from Hiroshima University, Hiroshima, Japan, in 1960.

He taught mathematics for two years at a high school in Osaka, Japan, and since 1962 has been a Research Assistant in the Department of Electronics, University of Osaka Prefecture, Osaka. His current research interests lie in the areas of digital systems and computer arithmetics.

Mr. Kinoshita is a member of the Institute of Electrical Engineers of Japan and the Information Processing Society of Japan.



Hideo Kosako (S'58–M'62) was born in Osaka, Japan, on November 15, 1930. He received the B.E. and M.E. degrees in electrical engineering from the University of Osaka Prefecture, Osaka, Japan, in 1953 and 1957, respectively, and the Ph.D. degree in electrical communication engineering from Osaka University, Osaka, in 1960.

He joined the faculty of Osaka University in 1960. Since 1961 he has been an Associate Professor in the Department of Electronics, University of Osaka Prefecture. During 1968-1969 he was a Visiting Professor of Electrical Engineering at the University of Arizona, Tucson. His teaching and research interests include hybrid computing systems and simulations.

Dr. Kosako is a member of the Institute of Electronics and Communication Engineers of Japan.

Yoshiaki Kojima (SM'57) was born in Osaka, Japan, on September 29, 1923. He received the B.E. and Ph.D. degrees in electrical engineering from the Tokyo Institute of Technology, Tokyo, Japan, in 1946 and 1961, respectively.

From 1946 to 1950 he was with the Sharp Corporation, Japan. In 1950 he joined the staff of the University of Osaka Prefecture, Osaka, and since 1963 he has been a Professor in the Department of Electronics. His present research interests include the areas of logical operation systems.

Dr. Kojima is a member of the Institute of Electronics and Communication Engineers of Japan, the Information Processing Society of Japan, and the Institute of Electrical Engineers of Japan.