# AN IMPROVEMENT OF FAST BASE EXTENSION IN RNS

F. Barsi and M.C. Pinotti

Nota interna B4-05

Febbraio 1992

Servizio Tecnografico - Area di Ricerca del CNR - Pisa

# AN IMPROVEMENT OF FAST BASE EXTENSION IN RNS†

F. Barsi* and M.C. Pinotti*

† This work has been supported by the C.N.R. project "Sistemi Informatici e Calcolo Parallelo";

* Istituto di Elaborazione dell'Informazione, Consiglio Nazionale delle Ricerche, Via S. Maria 46, 56100 Pisa, Italy.

**Abstract.**

New techniques to evaluate base extension have been previuosly presented in the literature allowing faster computation times. These techniques are based on the Chinese Remainder Theorem and are implemented by using lookup tables. However, the proposed methods require a sligth redundancy to be added to the number representation. In this paper, it is shown that this redundancy can be avoided if the terms of the Chinese Remander Theorem are properly expanded. The average base extension execution time is again bounded by log n table lookup cycles whereas area is sligthly increased.

**Index terms**

Chinese Remainder Theorem, Computer Arithmetic, Intermodular Operations, Mixed Radix Conversion, Residue Number Systems, Table lookup technique

## 1. Introduction

Arithmetic operations, based on RNS, have received renewed attention for many signal processing implementations. Consequently, the efforts to speed up intermodular operations (whose final or intermediate step is a conversion of numbers to a weighted notation) have been growing in the last years.

Base extension is perhaps the most important intermodular operation. It is necessary whenever the range of the representation is to be expanded to include additional moduli or a different RNS system is to be considered and can be used for performing sign or magnitude determination or scaling. The base extension procedure consists essentially in a conversion into a weighted number system followed by a mod m computation of the previous result.

There are two method allowing converting an RNS representation into a weighted one: the mixed radix conversion algorithm (MRC) reported in [1] and working in the modular ranges of the original RNS and the Chinese Remainder Theorem (CRT) which operate in the full range of definition of numbers. The first method requires (n - 1) multiplicative - additive steps. It has been preferred since now for base extension as a single, additional step is required to obtain the result after MRC process; however, additional moduli must be pairwise prime with the moduli of the original RNS system. The Chinese Remainder Theorem implies that computation are performed in the full range M of number definition and has as a final step a mod M computation.

In an original paper [2], Shenoy and Kumaresan proposed appending a redundant information to the RNS representation of numbers with the aim of escaping mod M operation in CRT. In fact, any residue representation $\{x_1, x_2, \ldots, x_{n-1}, x_n\}$ in the RNS of moduli $m_1, m_2, \ldots, m_{n-1}, m_n$ corresponds to the integer X:

$$X = \left| \Sigma_{i=1}^{n} M_i \left| \frac{x_i}{M_i} \right|_{m_i} \right|_M = \Sigma_{i=1}^{n} M_i \left| \frac{x_i}{M_i} \right|_{m_i} - k\,M \qquad (1)$$

where M is the product of moduli, $M_i = M/m_i$ and $|1/M_i|_{m_i}$ is the multiplicative inverse of $M_i$ mod $m_i$ and k is an integer in the range [0, n-1]. In general, constant k is unknown. However, Shenoy and Kumaresan showed that it can be derived from a redundant information consisting of an additional residue digit with respect to a redundant modulus $m_{n+1}$ by means of a procedure requiring $\lceil \log (n+1) \rceil$ table lookup cycles. When used for base extension, this method requires $\lceil \log (n+1) \rceil + 1$ table lookup cycles.

In this paper it will be shown that the same time performances can be obtained, without any redundancy, by means of an appropriate decompositions of CRT terms and the same approach can be exploited in RNS exact scaling.

## 2. A CRT decomposition

Let an RNS be defined by a set of pairwise prime moduli $\{m_1, m_2, \dots, m_{n-1}, m_n\}$ and let :

$$X \equiv \{x_1, x_2, \dots, x_{n-1}, x_n\} \qquad x_i = |X|_{m_i}, \quad i = 1, 2, \dots, n \tag{2}$$

be any integer in the range [0, M), $M = \Pi_{i=1}^{n} m_i$. Performing base extension with respect to a modulus $m_{n+1}$ starting from Chinese Remainder Theorem is equivalent to computing expression:

$$x_{n+1} = \left|\Sigma_{i=1}^{n} M_i \left|\frac{x_i}{M_i}\right|_{m_i}\right|_M \Bigg|_{m_{n+1}} = \left|\Sigma_{i=1}^{n} M_i \left|\frac{x_i}{M_i}\right|_{m_i} - k\,M\right|_{m_{n+1}} \tag{3}$$

Unfortunately, computations are to be carried out in the range M as the summation:

$$\Sigma_{i=1}^{n} M_i \left|\frac{x_i}{M_i}\right|_{m_i}$$

may exceed many times the range M and k is unknown. To avoid these obstacles, a method [2] has been proposed requiring a redundant modulus to be added to residue representation. This method consistently enhances the computation times. However, it will be shown that similar results can be obtained, without any redundancy, by means of an approach which has been suggested in [ ] for approximate scaling.

Letting, in general, $M_{i,j,\dots,t} = M/m_i m_{j\dots,} m_t$, observe that the projection of X on $M/m_n$ can be expressed as:

$$|X|_{M_n} \equiv \{x_1, x_2, \dots, x_{n-1}\} \tag{4}$$

and, applying CRT:

$$|X|_{M_n} = \left|\Sigma_{i=1}^{n-1} M_{i,n} \left|\frac{x_i}{M_{i,n}}\right|_{m_i}\right|_{M_n} \tag{5}$$

From the Fundamental Theorem of the Remainder and recalling that $|AX|_{AB} = A|X|_B$:

$$M_{i,n}\left|\frac{x_i}{M_{i,n}}\right|_{m_i} = a_i\, M_{n-1,n} + \left|M_{i,n}\left|\frac{x_i}{M_{i,n}}\right|_{m_i}\right|_{M_{n-1,n}} = a_i\, M_{n-1,n} + M_{i,n-1,n}\left|m_{n-1}\left|\frac{x_i}{M_{i,n}}\right|_{m_i}\right|_{m_i} \tag{6}$$

$$a_i = \left\lfloor \frac{M_{i,n}\left|\frac{x_i}{M_{i,n}}\right|_{m_i}}{M_{n-1,n}} \right\rfloor = \left\lfloor \frac{m_{n-1}}{m_i}\left|\frac{x_i}{M_{i,n}}\right|_{m_i} \right\rfloor \tag{6'}$$

Substituting in (5) , and observing that:

$$M_{i,n-1,n}\left|m_{n-1}\left|\frac{x_i}{M_{i,n}}\right|_{m_i}\right|_{m_i} = \left|M_{i,n}\left|\frac{x_i}{M_{i,n}}\right|_{m_i}\right|_{M_{n-1,n}} \equiv 0$$

for i = n-1, it is obtained:

$$|X|_{M_n} = \left|\Sigma_{i=1}^{n-1}\left(a_i\, M_{n-1,n} + M_{i,n-1,n}\left|m_{n-1}\left|\frac{x_i}{M_{i,n}}\right|_{m_i}\right|_{m_i}\right)\right|_{M_n} =$$

$$= \left| M_{n-1,n}\,\Sigma_{i=1}^{n-1} a_i + \Sigma_{i=1}^{n-2} M_{i,n-1,n}\left|m_{n-1}\left|\frac{x_i}{M_{i,n}}\right|_{m_i}\right|_{m_i}\right|_{M_n} = \qquad (7)$$

$$= \left| M_{n-1,n}\left|\Sigma_{i=1}^{n-1} a_i\right|_{m_{n-1}} + \Sigma_{i=1}^{n-2} M_{i,n-1,n}\left|m_{n-1}\left|\frac{x_i}{M_{i,n}}\right|_{m_i}\right|_{m_i}\right|_{M_n}$$

The mod $M_n$ argument:

$$X_{E,M_n} = M_{n-1,n}\left|\Sigma_{i=1}^{n-1} a_i\right|_{m_{n-1}} + \Sigma_{i=1}^{n-2} M_{i,n-1,n}\left|m_{n-1}\left|\frac{x_i}{M_{i,n}}\right|_{m_i}\right|_{m_i} \qquad (8)$$

has the following limitations:

$$0 \le X_{E,M_n} \le M_{n-1,n}\,(m_{n-1}-1) + M_{1,n-1,n}\,(m_1-1) + M_{2,n-1,n}\,(m_2-1) + \ldots + M_{n-2,n-1,n}\,(m_{n-2}-1) =$$
$$= M_n + (n-3)\,M_{n-1,n} - M_{n-1,n}\left(\frac{1}{m_1} + \frac{1}{m_2} + \ldots + \frac{1}{m_{n-2}}\right) < 2\,M_n \qquad (8')$$

and the $M/m_n$ projection becomes:

$$|X|_{M_n} = X_{E,M_n} - \varepsilon_{M_n} M_n \qquad \varepsilon_{M_n} = 0,\ 1 \qquad (9)$$

Observing that X, $|X|_{M_n}$ and $X_{E,M_n}$ differ for a multiple of $M_n$ , it follows that:

$$X = X + X_{E,M_n} - X_{E,M_n} = X_{E,M_n} + M_n\left\lfloor\frac{X - X_{E,M_n}}{M_n}\right\rfloor \qquad (10)$$

and, mod M:

$$|X|_M = X = \left|X_{E,M_n} + M_n\left\lfloor\frac{X - X_{E,M_n}}{M_n}\right\rfloor_M\right| = \left|X_{E,M_n} + M_n\left\lfloor\frac{X - X_{E,M_n}}{M_n}\right\rfloor_{m_n}\right|_M \qquad (11)$$

Recalling (8'), the argument:

$$X_E = X_{E,M_n} + M_n\left\lfloor\frac{X - X_{E,M_n}}{M_n}\right\rfloor_{m_n} = X_{E,M_n} + M_n\left|\frac{1}{M_n}\right|_{m_n}\left(x_n - |X_{E,M_n}|_{m_n}\right)\Big|_{m_n} \qquad (12)$$

will range in:

$$\left[0,\ M + (n\text{-}3)\,M_{n-1,n} - M_{n-1,n}\left(\frac{1}{m_1} + \frac{1}{m_2} + \ldots + \frac{1}{m_{n-2}}\right)\right] \qquad (12')$$

Consequently:

$$X = X_E - \varepsilon M = X_{E,M_n} + M_n \left| \frac{X - X_{E,M_n}}{M_n} \right|_{m_n} - \varepsilon M \qquad \varepsilon = 0, 1 \tag{13}$$

and $\varepsilon = 1$ if and only if:

$$X_E \geq M \tag{14}$$

Substituting for $X_E$ in (14) the above condition becomes:

$$M_{n-1,n} \left| \Sigma_{i=1}^{n-1} a_i \right|_{m_{n-1}} + \Sigma_{i=1}^{n-2} M_{i,n-1,n} \left| m_{n-1} \left| \frac{x_i}{M_{i,n}} \right|_{m_i} \right|_{m_i} + M_n \left| \frac{1}{M_n} \right|_{m_n} (x_n - |X_{E,M_n}|_{m_n}) \Big|_{m_n} \geq M \tag{14'}$$

and, dividing by $M_{n-1,n}$ both sides of the inequality and rearranging terms:

$$\left| \Sigma_{i=1}^{n-1} a_i \right|_{m_{n-1}} + m_{n-1} \left| \left| \frac{1}{M_n} \right|_{m_n} (x_n - |X_{E,M_n}|_{m_n} \right|_{m_n} \geq m_n m_{n-1} - \Sigma_{i=1}^{n-2} \frac{1}{m_i} \left| m_{n-1} \left| \frac{x_i}{M_{i,n}} \right|_{m_i} \right|_{m_i} \geq$$

$$\geq m_n m_{n-1} - (n - 2) + \Sigma_{i=1}^{n-2} \frac{1}{m_i} \geq m_n m_{n-1} - (n - 3) \tag{15}$$

Inequality (15) shows a necessary condition for $\varepsilon = 1$. The same condition can be given the following equivalent formulation:

$$T_1 = \left| \Sigma_{i=1}^{n-1} a_i \right|_{m_{n-1}} \geq m_{n-1} - (n - 3) \tag{15'}$$

$$T_2 = \left| \left| \frac{1}{M_n} \right|_{m_n} (x_n - |X_{E,M_n}|_{m_n}) \right|_{m_n} = m_n - 1 \tag{15''}$$

Now, suppose that conditions (15') and (15") are satisfied. Then:

$$X_E = M_n T_2 + M_{n-1,n} T_1 + \Sigma_{i=1}^{n-2} M_{i,n-1,n} \left| m_{n-1} \left| \frac{x_i}{M_{i,n}} \right|_{m_i} \right|_{m_i} \geq$$

$$\geq M_n (m_n - 1) + M_{n-1,n} (m_{n-1} - (n - 3) = M - (n - 3) M_{n-1,n}$$

and, recalling (12'):

$$M - (n - 3) M_{n-1,n} \leq X_E \leq M + (n - 3) M_{n-1,n} - M_{n-1,n} \Sigma_{i=1}^{n-2} \frac{1}{m_i} \tag{16}$$

or, equivalently:

$$|X_E|_M = X \geq M - (n - 3) M_{n-1,n}$$

if $M - (n - 3) M_{n-1,n} \leq X_E < M$, i.e., $\varepsilon = 0$ and

$$|X_E|_M = X \le (n-3)\, M_{n-1,n} - M_{n-1,n} \Sigma_{i=1}^{n-2} \frac{1}{m_i} \qquad (17)$$

if $M \le X_E \le M + (n-3)\, M_{n-1,n} - M_{n-1,n} \Sigma_{i=1}^{n-2} \frac{1}{m_i}$, i.e., $\varepsilon = 1$.

Preceding considerations represent the proof to the following Lemma.

***Lemma 1.*** In an RNS of moduli $m_1, m_2, \ldots, m_{n-1}, m_n$, any representation $X \equiv \{x_1, x_2, \ldots, x_{n-1}, x_n\}$ corresponds to the integer

$$X = X_{E,M_n} + M_n \left| \frac{X - X_{E,M_n}}{M_n} \right|_{m_n} - \varepsilon M = X_{E,M_n} + M_n \left| \frac{1}{|M_n|_{m_n}} (x_n - |X_{E,M_n}|_{m_n}) \right|_{m_n} - \varepsilon M \qquad (13)$$

where:

$$X_{E,M_n} = M_{n-1,n} \left| \Sigma_{i=1}^{n-1} a_i \right|_{m_{n-1}} + \Sigma_{i=1}^{n-2} M_{i,n-1,n} \left| m_{n-1} \left| \frac{x_i}{|M_{i,n}|_{m_i}} \right|_{m_i} \right|_{m_i} \qquad (8)$$

$$a_i = \left\lfloor \frac{M_{i,n} \left| \frac{x_i}{|M_{i,n}|_{m_i}} \right|}{M_{n-1,n}} \right\rfloor = \left\lfloor \frac{m_{n-1}}{m_i} \left| \frac{x_i}{|M_{i,n}|_{m_i}} \right| \right\rfloor \qquad (6')$$

and $\varepsilon = 0, 1$.

$\varepsilon = 1$ if and only if the following conditions simultaneously hold:

$$T_1 = \left| \Sigma_{i=1}^{n-1} a_i \right|_{m_{n-1}} \ge m_{n-1} - (n-3) \qquad (15')$$

$$T_2 = \left| \left| \frac{1}{|M_n|_{m_n}} (x_n - |X_{E,M_n}|_{m_n}) \right|_{m_n} = m_n - 1 \right. \qquad (15'')$$

$$X \le (n-3)\, M_{n-1,n} - M_{n-1,n} \Sigma_{i=1}^{n-2} \frac{1}{m_i} \qquad (17)$$

## 3. Base extension

Consider the RNS of pairwise prime moduli $\{m_1, m_2, \ldots, m_{n-1}, m_n\}$ and an integer X represented in the given system as:

$$X \equiv \{x_1, x_2, \ldots, x_{n-1}, x_n\}$$

Let $m_{n+1}$ be any positive integer and suppose we wish to extend the residue representation $\{x_1, x_2, \ldots, x_{n-1}, x_n\}$ to include $m_{n+1}$ as an additional modulus. From Lemma 1, $x_{n+1}$ can be evaluated as:

$$x_{n+1} = \left| M_{n-1,n}\, T_1 + M_n\, T_2 + \Sigma_{i=1}^{n-2} M_{i,n-1,n} \right|_{m_{n-1}} \left| \frac{x_i}{M_{i,n}} \right|_{m_i} \Big|_{m_i} - \varepsilon M \Big|_{m_{n+1}} \qquad (18)$$

where $\varepsilon$ is determined according to conditions of Lemma. The first step consists in computing $T_1$ and $T_2$ as these terms appear in (18) and in conditions (15') and (15").

Assuming that a table lookup technique is used to carry out arithmetic, $T_1$ can be be computed according to the logical scheme reported in Figure 1.

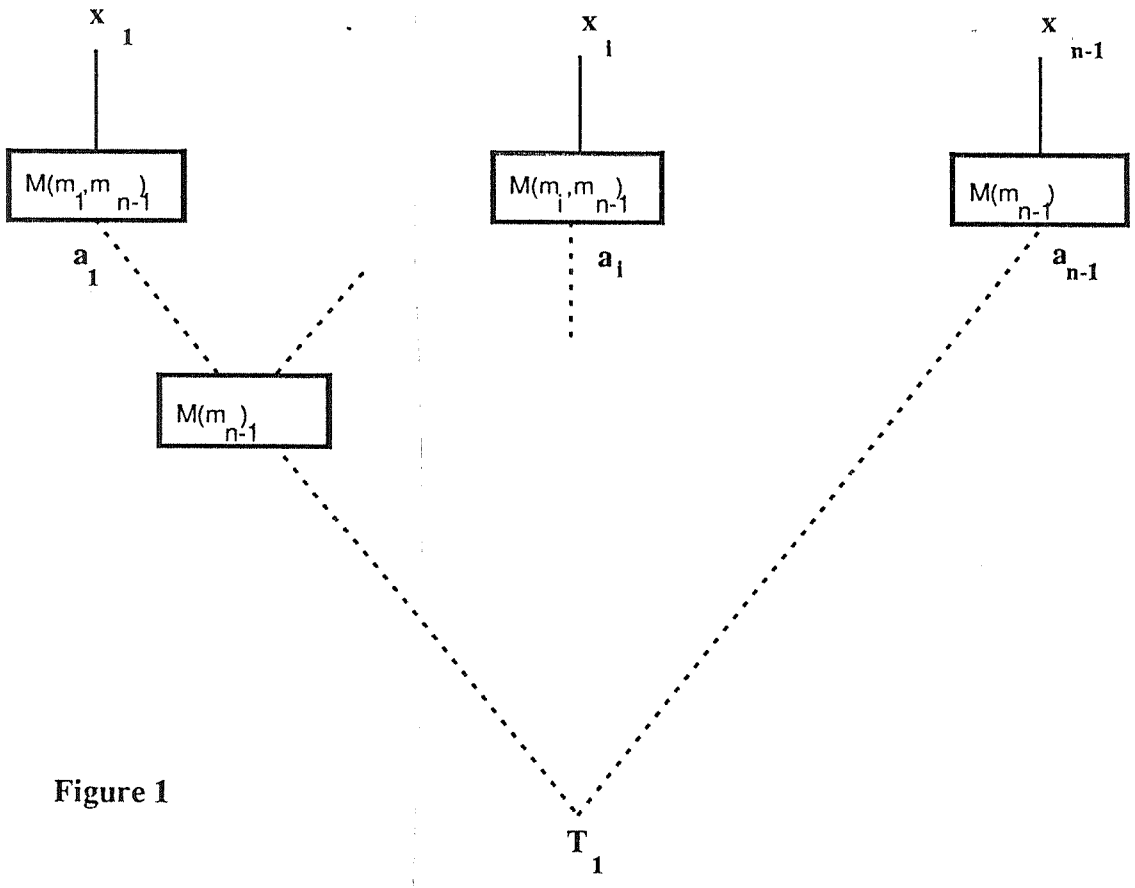


**Figure 1**

Following this scheme, each term $a_i$ (6') is obtained by addressing a memory $M(m_i, m_{n-1})$ with $m_i$ words of $\log m_{n-1}$ bits. Starting from $a_i$'s, $T_1$ is computed by a tree of $m^2_{n-1} \times \log m_{n-1}$ bit memory elements, outlined as $M(m_{n-1})$.

Evaluating $T_2$ is a little more complicated as the extended expression becomes:

$$T_2 = \left| \frac{1}{M_n} \right|_{m_n} (x_n - |X_{E,M_n}|_{m_n}) \Big|_{m_n} = \left| \frac{1}{M_n} \right|_{m_n} (x_n - \left| M_{n-1,n} \Sigma_{i=1}^{n-1} a_i \right|_{m_{n-1}} + \Sigma_{i=1}^{n-2} M_{i,n-1,n} \Big|_{m_{n-1}} \left| \frac{x_i}{M_{i,n}} \right|_{m_i} \Big|_{m_i} \Big|_{m_n} ) \Big|_{m_n} =$$

$$= \left| \frac{1}{M_n} \right|_{m_n} (x_n - \left| M_{n-1,n} \Sigma_{i=1}^{n-1} a_i \right|_{m_{n-1}} \Big|_{m_n} - \Sigma_{i=1}^{n-2} \left| M_{i,n-1,n} \right|_{m_{n-1}} \left| \frac{x_i}{M_{i,n}} \right|_{m_i} \Big|_{m_i} \Big|_{m_n} ) \Big|_{m_n}$$

Letting, for simplicity:

$$A = |M_{n-1,n}\ T_1|_{m_n}$$

$$B = \left|\Sigma_{i=1}^{n-2} \left|M_{i,n-1,n}\right|_{m_{n-1}} \left|\frac{x_i}{M_{i,n}}\right|_{m_i}\right|_{m_n}$$

the computation of $T_2$ is carried out as sketched in Figure 2.

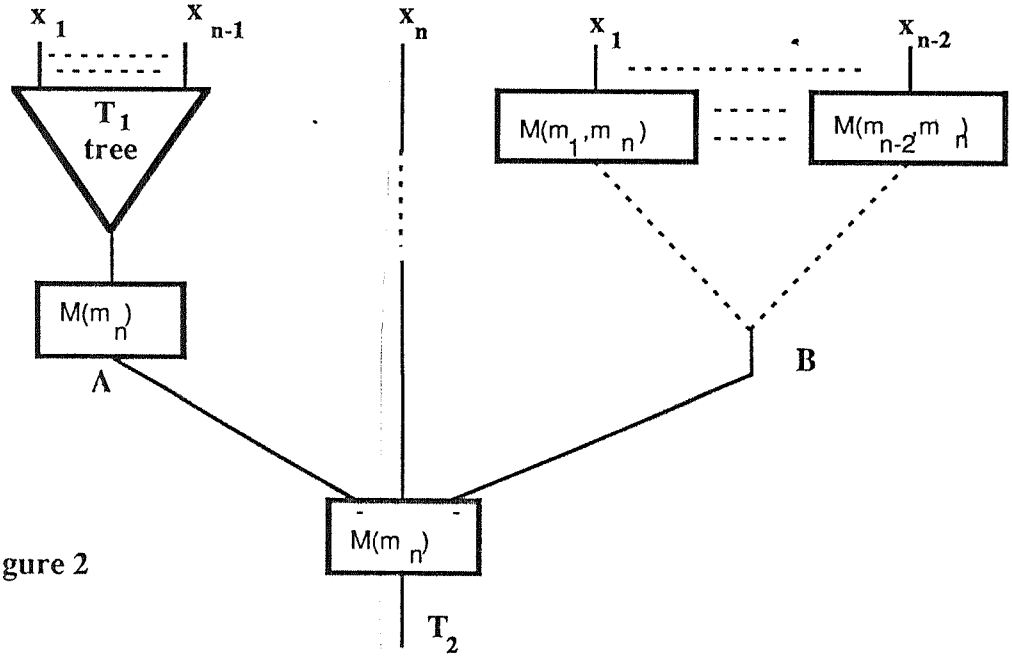


Figure 2

Here, B is obtained by using a tree of $m^2_n$ x log $m_n$ bit memory elements, whereas A is obtained from $T_1$ in a single memory cycle.

Once inequalities (15') and (15") of Lemma 1 are satisfied (and $X_E$ ranges in (16)), the last condition (17) is equivalent to verify if X is in the range $[0, M_n)$ or, in other terms:

$$|X|_{M_n} = X$$

To do this, an application of MRC algorithm is necessary to verify if the most significant digit $x^*_n$ of the associate mixed radix representation of X is "0". The computational scheme for MRC is reported in Figure 3.

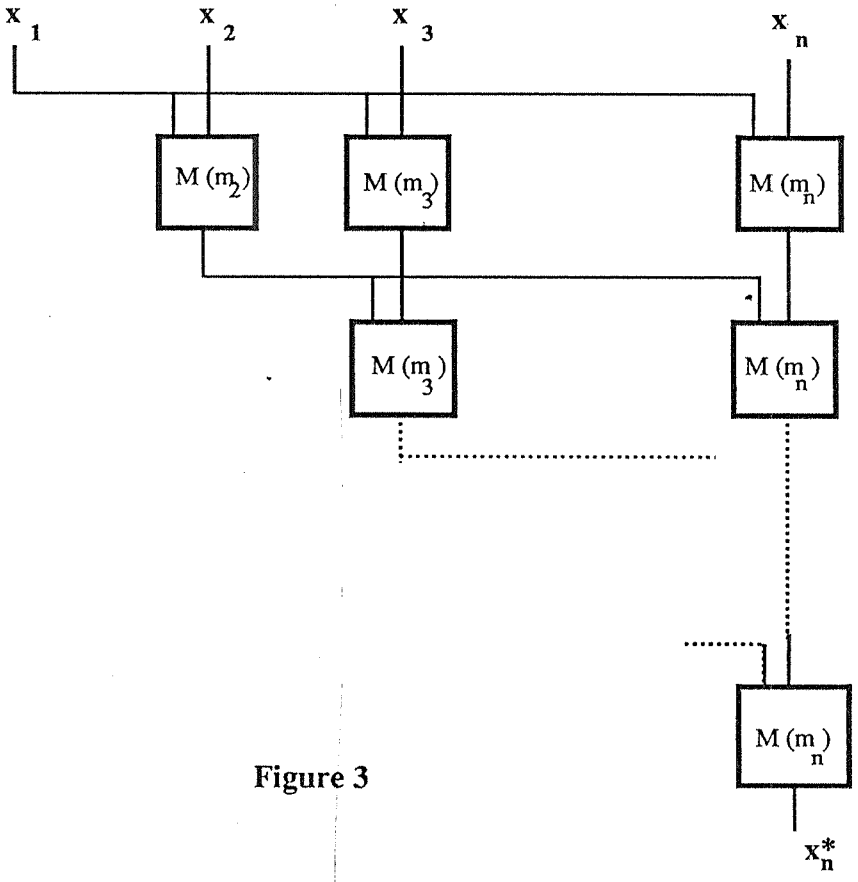


**Figure 3**

As a conclusion, the logical computational scheme for $x_{n+1}$ is shown in the following Figure 4, where:

$a = |M_{n-1,n}\ T_1|_{m_{n+1}}$

$b = \left|\Sigma_{i=1}^{n-2} \left| M_{i,n-1,n} \left| m_{n-1} \left| \frac{x_i}{M_{i,n}} \right|_{m_i} \right|_{m_i} \right|_{m_{n+1}} \right|_{m_{n+1}}$

$c = |M_n\ T_2|_{m_{n+1}}$

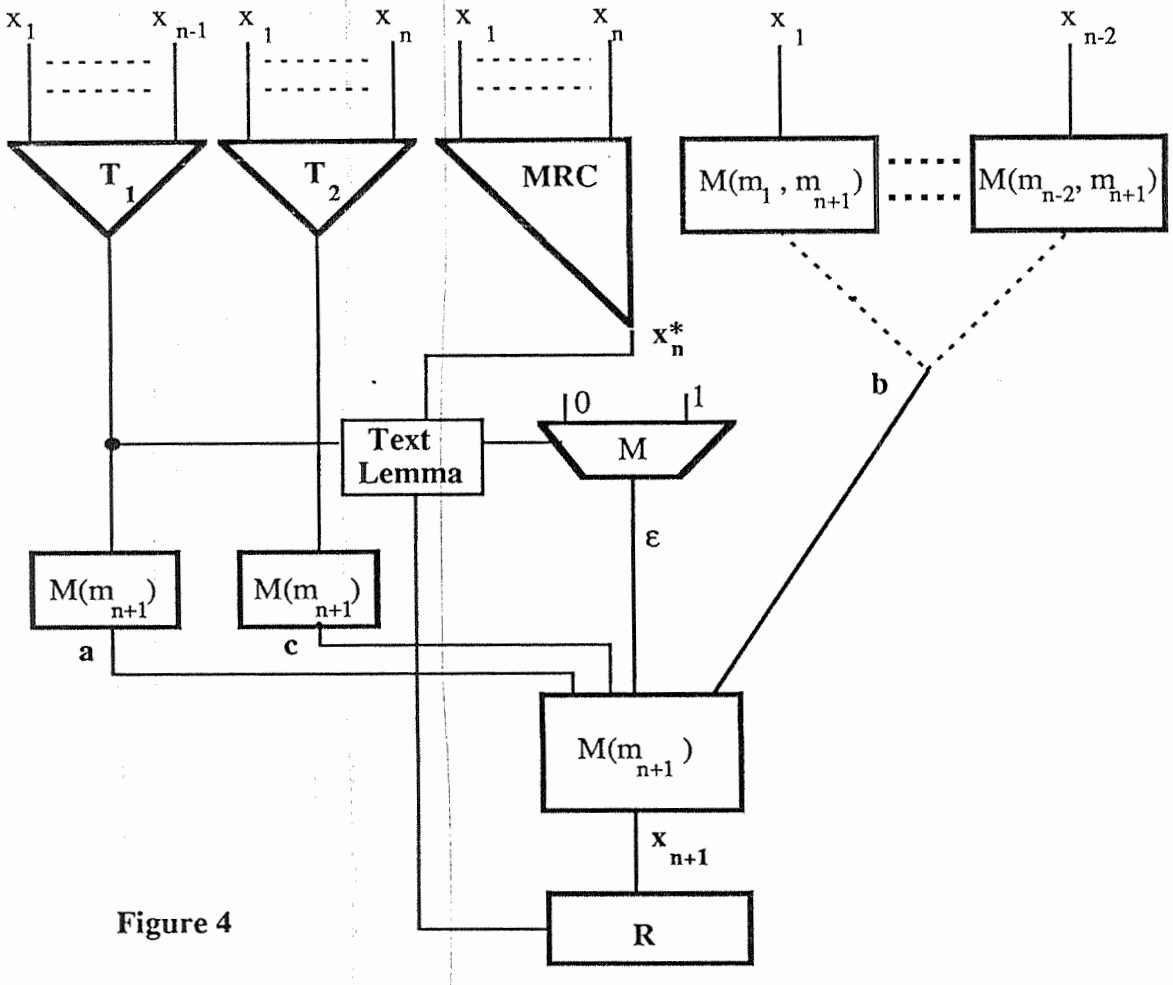


**Figure 4**

To completely realize the scheme presented in figure, it must be observed that conditions (15'), (15") and (17) of Lemma 1 are verified for (n - 3) out of $m_{n-1}\ m_n$ possible combinations of $T_1$ and $T_2$ . Consequently, in most cases, the result $x_{n+1}$ is available (and stored in register R) as soon as $T_1$ , $T_2$ and "b" are computed, i.e., after log n memory cycles. On the contrary, if all conditions of Lemma are verified, the result is obtained (and stored) only at the end of MRC process, which requires n memory cycles. The average time of computation is then:

$$t_{AV} = \frac{n-3}{m_{n-1}\ m_n}\, n + \frac{m_{n-1}\ m_n - (n-3)}{m_{n-1}\ m_n} \log n$$

memory cycles.

Before going on, let's evaluate time performances and area occupancy of a VLSI structure implementing the proposed algorithm. To this purpose, it will be assumed that:

a) $m_i = \theta\ (m)$, $i = 1, 2, \ldots , n+1$

b) $\log m = \theta (n)$

c) memory modules are organized following a hierarchical scheme.

In a hierarchical organization [4] [5], the access time and the area occupancy of a two operands, mod m, memory module can be expressed (see Figure 5) as:

$$t_M = \max \{N a_0 \gamma, N a_0 \beta\} \quad (19)$$

$$A = H_M \; W_M \quad (20)$$

where $H_M = N + \gamma^N$ , $W_M = N + \beta^N$ and $\gamma^N \beta^N = m^2 \log m$

where N is the number of hierarchical levels, $a_0$ is a technology dependent constant and $\gamma$ and $\beta$ are the branching factors of memory trees.

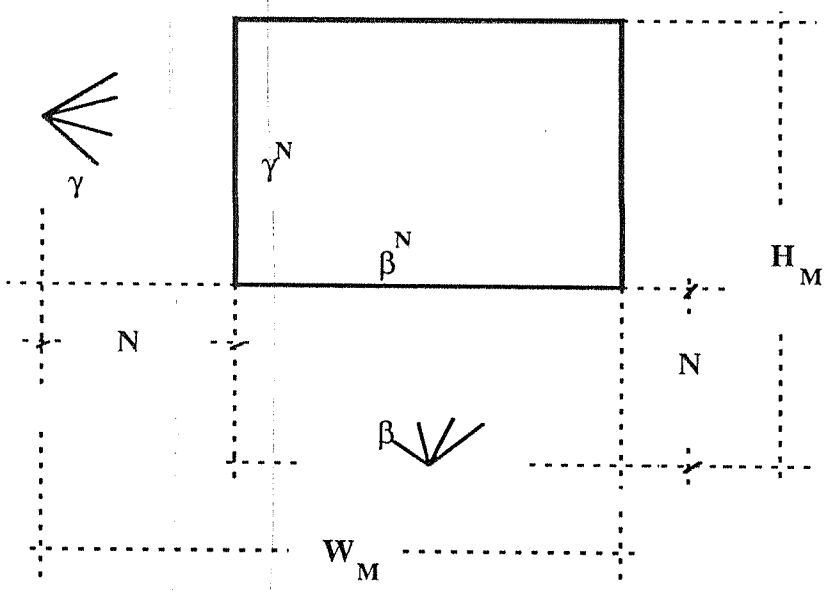


**Figure 5**

In the hypothesis that the major goal is minimizing the memory access time, it follows from (19) that $\gamma = \beta$ and, assuming that $\gamma$ is a constant, (19), (20) take the form:

$$t_M = t_0 N \quad (19')$$

$$\gamma^{2N} = m^2 \log m \quad (20')$$

and

$$N = \theta (\log m)$$

$t_M = \theta (\log m)$ (19")

$A_M = \theta (m^2 \log m)$ (20")

as $H_M = \theta (m \log^{1/2} m)$, $W_M = \theta (m \log^{1/2} m)$

Now, let's reconsider the computational schemes of Figures 1-4, recalling that look up tables are used to perform arithmetic.

Figure 1 and 2 show that $T_1$ and $T_2$ are computed by means of trees of memories. This observations are synthesized in the following Figure 6.

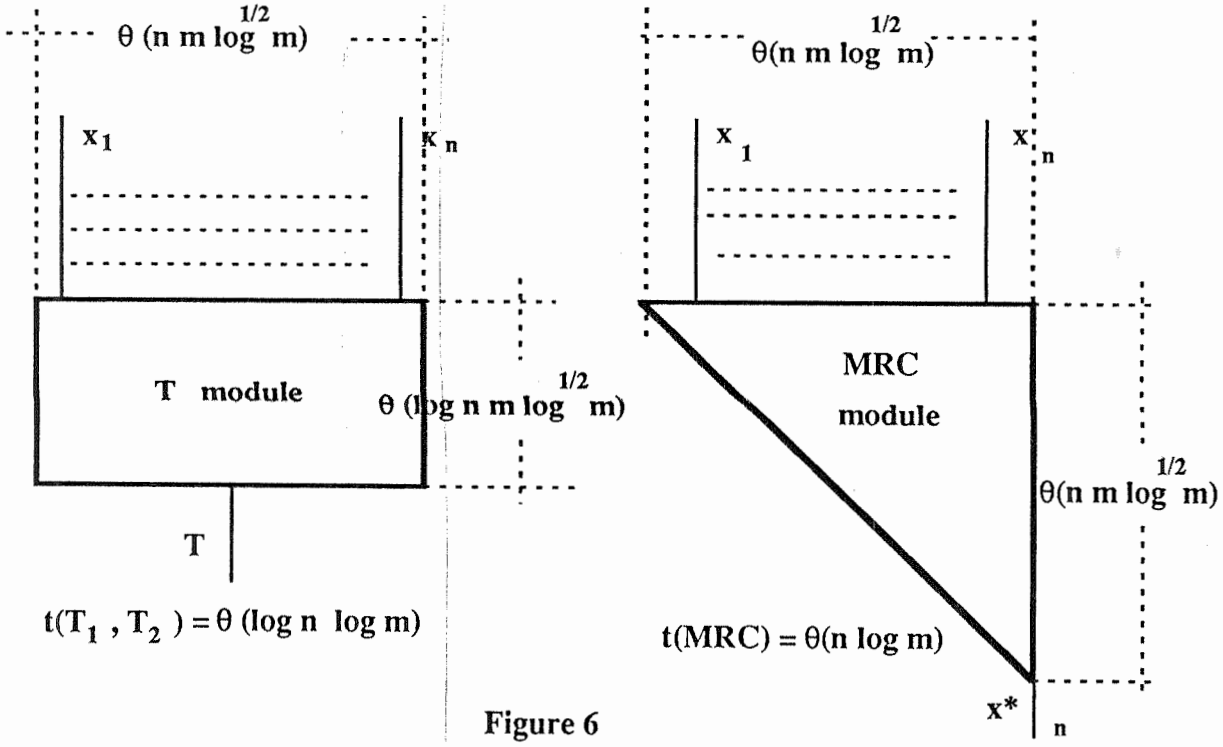


**Figure 6**

Preceding considerations make it possible to evaluate the overall time performances and area occupancy of a VLSI structure corresponding to the computational scheme of Figure 4. In fact, the average time to perform base extension can be expressed as:

$T_{EXT} = \theta (f_{MRC} \, n \log m + (1 - f_{MRC}) \log n \log m)$

where $f_{MRC}$ represents the probability of doing a mixed radix conversion. Recalling that MRC is required for (n-3) instances of the product $T_1 \, T_2$ out of $m_{n-1} \, m_n$ possible values, it follows:

$f_{MRC} = \theta (n/m^2)$

i.e.,

$$T_{EXT} = \theta\ ((n/m^2)\ n \log m + (1 - (n/m^2))\ \log n \log m) = \theta\ (\log n \log m)$$

which coincides with the time performances obtained in [2] by using redundancy. Similarly, deriving the area occupancy is trivial and results:

$$A_{EXT} = \theta\ (n^2\ m^2 \log m)$$

Note that, unfortunately, there is no real time savings with respect to the well known method based on MRC. Namely, every memory cycle takes time proportional to the length of the memory address, that is $\theta$ (log m), whereas each additive-multiplicative operation of MRC works in $\theta$ (log log m) time. With the previous assumptions a) and b), the total time $\theta$ (n log log m) of MRC procedure and $\theta$ (log n log m) of the proposed method coincide. However, a different realization of the proposed CRT decomposition could be investigated using a tradeoff between adders, multipliers and lookup tables to minimize the total time of the base extension operation.

## 4. Scaling by a product of moduli

In preceding Section 3 we have shown how high time performances for base extension can be reached without any redundancy request. However, the possibilities of the proposed method are best exploited in applications such as scaling where exact results are obtained in $\theta$(log n) memory cycles. In the literature, the same time results have been obtained at the expenses of a slight approximation [3].

In the RNS of pairwise prime moduli $\{m_1, m_2, \ldots, m_{n-1}, m_n\}$ let X be an integer represented as:

$$X \equiv \{x_1, x_2, \ldots, x_{n-1}, x_n\}$$

and suppose we wish to scale X by a scaling factor $F_s$ equal to the product of the s lowest order moduli:

$$F_s = \Pi_{i=1}^{s}\ m_i$$

The scaled integer:

$$X_s = \left|\frac{X}{F_s}\right| = \frac{X - |X|_{F_s}}{F_s} = \left|\frac{X - |X|_{F_s}}{F_s}\right|_{\frac{M}{F_s}} = \left|\frac{1}{F_s}\right|_{\frac{M}{F_s}} \left(|X|_M - |X|_{F_s}|_M\right)\Big|_{\frac{M}{F_s}} \qquad (21)$$

is representable in the range $[0, M/F_s)$ and is obtained by means of a base extension of:

$$|X|_{F_s} \equiv \{x_1, x_2, \ldots, x_s\}$$

to the range $M/F_s$ , i.e., to the moduli $m_{s+1}, \ldots, m_n$. Once $X_s$ is computed in the reduced range, a final base extension is required to obtain the full range representation. The computational scheme is shown in Figure 7.

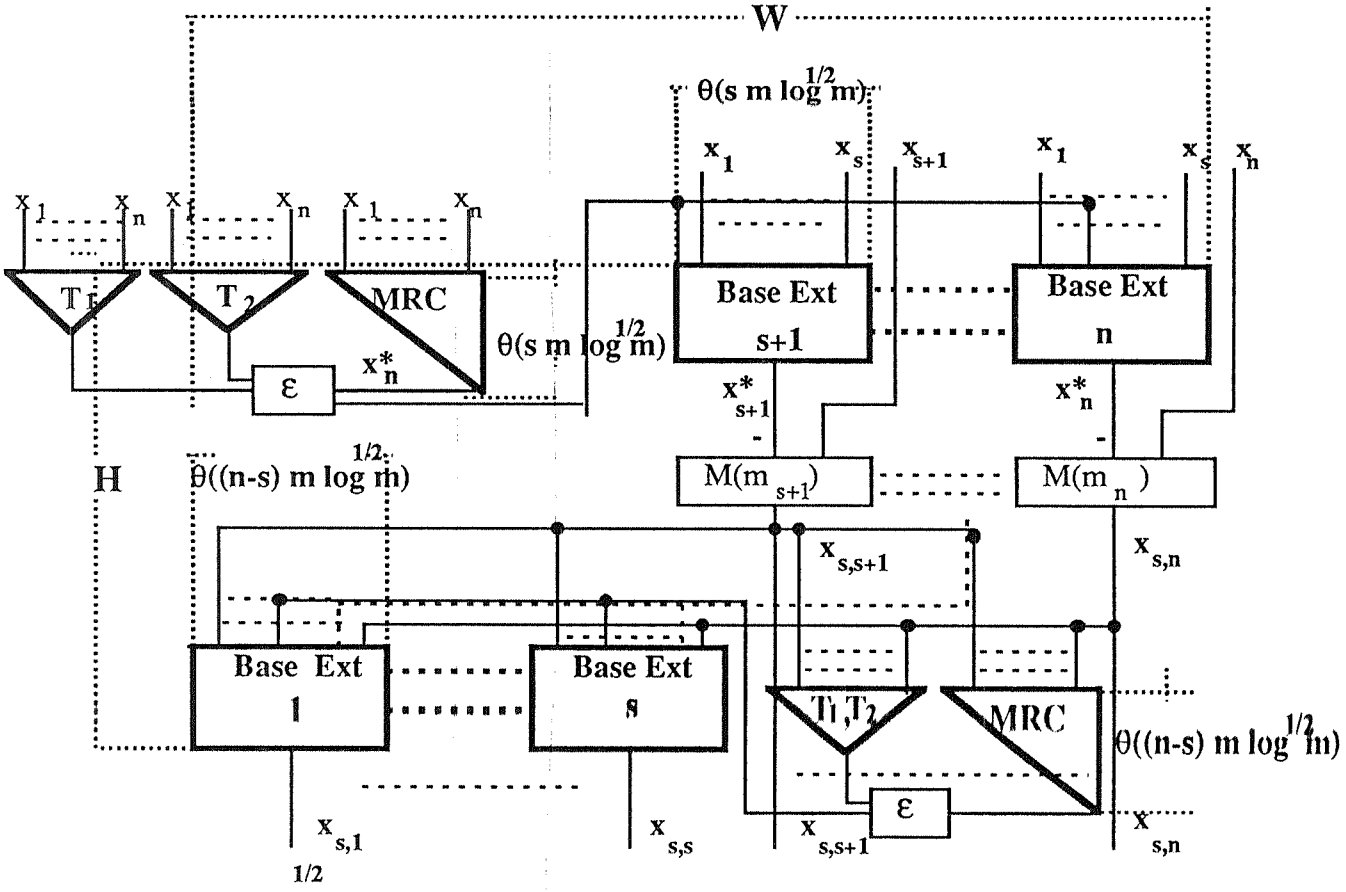


**Figure 7**

Note that ε depends only on x and does not depend on the current extension. Then it is calculated once to extend $|X|_{F_s}$ and once to obtain the final full range representation.

The average time required to perform the first part of scaling,i.e., extending $|X|_{F_s}$ to moduli $m_{s+1}, m_{s+2}, \ldots, m_n$ in order to obtain $\{x^*_{s+1}, x^*_{s+2}, \ldots, x^*_n\}$ and then $\{x_{s,s+1}, x_{s,s+2}, \ldots, x_{s,n}\}$ results:

$t_{S'} = \theta$ (log s log m)

The final part of the algorithm is again a base extension of $\{x_{s,s+1}, x_{s,s+2}, \ldots, x_{s,n}\}$ to the range $F_s$ for which:

$t_{S''} = \theta$ (log (n-s) log m)

To conclude:

$t_S = \theta$ (log m max{log s, log(n-s)})

whereas the area is:

$A_S = W \times H = \theta$ (m $\log^{1/2}$ s (n-s)) x $\theta$ ((m $\log^{1/2}$ m max {s, (n-s)}) =

$= \theta$ ($m^2$ log m n s (n-s)).

## Conclusion

A new CRT decomposition has been devised which allows to perform base extension in average $\theta$ (log n) table lookup cycles without redundancy.

Moreover, the possibilities of the proposed method are best exploited in scaling operation where the exact result is obtained in $\theta$ (log n) table lookup cycles.

## References


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

[2] A. P. Shenoy and R. Kumaresan, "Fast base exstension using a redundant modulus in RNS", *IEEE Trans. Comput.*, vol. 38, pp. 292-297, Feb 1989.

[3] A. P. Shenoy and R. Kumaresan, "A fast and accurate RNS scaling technique for high speed signal processing", *IEEE Trans. on Acoustic, Speech, and Signal Processing*, vol. 37, pp. 929-937, June 1989.

[4] C. A. Mead and M. Rem, "Cost and Performance of VLSI Computing Structures", *IEEE J. Solid-State Circuits*, vol. SC-14, pp. 453-462, Apr. 1979.

[5] M. Y. Bayoumi, G. A. Jullien and W. C. Miller, "A Look-up table VLSI Design Methodology for RNS Structures Used in DSP Applications", *IEEE Tans. on Circuits and Sistems*, vol. CAS-34, pp. 604-615, June 1987.