ON THE EFFICIENT IMPLEMENTATION OF HIGHER RADIX SQUARE ROOT ALGORITHMS

Paolo Montuschi and Luigi Ciminiera

Dipartimento di Automatica e Informatica,
Politecnico di Torino, corso Duca degli Abruzzi 24,
10129 Torino (Italy)

Abstract

Square root non-restoring algorithms operating with a radix higher than two (but power of 2) are discussed. Formulae are derived delimiting the feasibility space of the class of algorithms considered as a function of the different parameters. This definition leads to the determination of some of these parameters; in particular, it is possible to compute the number of partial remainder bits to be inspected for digit selection and the number of operand bits to be inspected to generated the first radicand value, as both parameters have a relevant impact on the implementation. Finally, the specific case of radix 4, digit set $\{-2, -1, 0, +1, +2\}$ and partial remainder represented by the sum of 2 numbers is considered.

1 Introduction

The possibility of implementing an ever increasing number of devices in a single integrated circuit has attracted attention to the design of arithmetic units working with a radix higher than 2. In fact, the additional hardware made available by technological advances may be used to increase the speed of arithmetic circuits, by performing computations using higher radix representations. An example of this trend is the INMOS T800 transputer [9], which integrates on a single chip both the processor and a floating point unit performing radix 8 multiplication and radix 4 division.

Square root has received less attention than the other basic arithmetic operations, because it is used less frequently; nevertheless, square rooting is mandatory for the implementation of the IEEE 754 floating point standard [10]. One of the first definitions of a non-restoring algorithm [1], [14] for square rooting may be found in [12]. More recently, square root algorithms for on-line arithmetic have been studied in [5], [6], [13]. These papers give some general formulae to achieve the convergence interval and digit selection. The algorithms presented are developed under the assumption that the digit set is maximal, that is, each digit ranges from $-(B-1)$ to $(B-1)$, where B is the radix. Square root computation for non-redundant operands is the subject of [11], where a thorough discussion and new high-speed algorithms for the radix 2 case are presented. Specific algorithms for higher radix square rooting are presented in [8] and [15]; the latter, in particular, uses a *non-canonical* digit set $\{-1, -1/2, 0, 1/2, 1\}$, so that the same hardware unit for both division and square root can be used. A square root/divide unit is also presented in [8], but using a radix 4 canonical digit set.

The present paper discusses the major aspects of non-restoring square root algorithm, with the operand represented with radix higher than 2 (but integral power of 2). In particular, this paper presents a set of formulae which are useful to evaluate design tradeoffs for the implementation of a square root extractor; the formulae presented are valid for any radix and digit set in the class considered. The only assumptions adopted are:

1. The value of the radicand X is represented in non-redundant form;
2. X belongs to the interval $[1/4, 1)$;
3. the square root value Y is represented in non-redundant form.

In section 2, a reference implementation of a square root unit is discussed to show the tradeoffs related to its implementation and to define all the parameters considered and their impact on the implementation. In section 3 the region of convergence and the intervals for digit selection are analyzed for a generic non-restoring algorithm. In section 4 a class of algorithms is presented, and the corresponding selection rules are derived. A short analysis of the problem of the determination of the lookup table is the subject of section 5. Finally a practical example of application for the case of radix 4 and digit set $[-2, +2]$ is provided in section 6.

2 Hardware unit

The algorithm for square root computation considered in this paper is based on the concept of completing the square:

$$X_i = X_{i-1} - (2Y_{i-1} + y_i B^{-i}) y_i B^{-i} = X_0 - Y_i^2$$

or in the form

$$X_i^* = BX_{i-1}^* - (2Y_{i-1} + y_i B^{-i}) y_i = (X_0 - Y_i^2) B^i \quad (1)$$

where

$$Y_i = Y_{i-1} + y_i B^{-i} \quad \text{and} \quad X_i^* = X_i B^i \quad (2)$$

Table 1 summarizes the symbols used and their definitions. Although the result digits belong to D_s, it is assumed that the partial result Y_i is always represented in non-redundant form, because the new digit produced at each step is input to an on-the-fly conversion mechanism [7] producing the corresponding irredundant representation of Y_i.

Note that the iteration index p and the number of bits i of the partially developed result are, in general, different (but correlated) entities. In particular, in this paper the computation of the square root value is carried out in two different phases. In the first phase an initialization procedure is performed by determining the most significant k bits of the result. Then, this is followed by the second phase, where, with a recursive process based on (1) and on the iteration index p, the subsequent digits of the result are computed. It is assumed that $p = 0$ during the initialization phase. It turns out that the relation between i and p is as follows $i = k/b + p$. As k is not necessarily an integer multiple of b, the value of i may be a rational number, however $b \cdot i$, which is the number of bits produced, is always integer. Observe that the non-integer values for i are caused by the initialization phase, which introduces a fixed shift in both the result and the partial remainder and this does not affect from a practical point of view the subsequent operations. In the rest of the paper, the algorithms will be analyzed only referring to the parameter i, since this choice allows a homogeneous notation to be used for both phases.

The initialization phase plays two different but very important roles in the whole algorithm. The first is to produce a sufficient number of fractional bits k_T of the result so that the corresponding shifted partial remainder belongs to the region of convergence. The second role is to provide a sufficient number of fractional bits k of Y_i to start the second phase of the algorithm using digit selection intervals independent of i. From the work in [3] it is known that, if the number of fractional bits to be produced during the first phase is k, both requirements are satisfied.

An unit carrying out the above specified algorithm may be implemented in several possible ways. However, in order to explain the impact on the implementation of the results presented in this paper, the structure shown in Fig. 1 will be taken as a reference model of the square root unit; it is worth noting that, among the possible models, this is one with the most popular implementations [8], [15] in parallel units. In Fig. 1, some of the most important symbols, as defined in Table 1, are indicated with their length in bits. The arithmetic core of the unit is the adder-subtractor together with the associated shifter by one digit position, which implement the recursion step (1). As the result of this operation is not strictly needed in irredundant form, the value of X_i^* is often obtained in a non-assimilated carry-sum form, in order to avoid full carry propagation through the adder. This advantage is paid for when it is necessary to select the new root digit, because the selection is based on the interval within which the new value of X_i^* lies within; as X_i^* is in redundant form, this operation requires the reduction to irredundant representation of the first r bits of X_i^*, by means of carry-look-ahead (CLA) adder. Of course it would be nice to keep r as small as possible, because it influences both the size of CLA and the size of the digit selection table (DST); on the other hand, the value of r depends on the representation of X_i^*, through ε_L and ε_H. The size of DST also depends on the number of bits, k, of the value of $\widehat{Y}$ determined through the lookup table (LT) during the initialization phase Also the value of k and r are correlated. Finally, the block DM performs the preparation of the second operand in the right side of (1), which basically requires the multiplication of Y_{i-1} by the new digit; this multiplication is often avoided by taking advantage of the small value of the multiplier. Although the digits $y_i \in D_s$, the block labeled Y_i in Fig. 1 is able to produce the partial and final results in irredundant form, by using the on-the-fly conversion mechanism [7].

The following sections will show how it is possible to relate the values of all the parameters of the unit in Fig. 1 discussed above; this analysis is valid for any radix, symmetrical signed digit set and representation of X_i^*. The

Table 1: Definitions of symbols used

p	iteration index; ($p \in N^+$)
i	number of digits of the square root value which have been computed
B	$= 2^b$; radix ($b > 0$ and integer)
X_i	partial remainder after the computation of i digits of the result
X_i^*	shifted partial remainder after the computation of i digits of the result; $X_i^* = X_i B^i$
X, X_0	radicand
N	number of bits of the radicand
D_S	$= \{-s, \ldots, 0, +1, \ldots, +s-1, +s\}$; digit set ($s$ is integer and $s \leq B-1$)
y_i	$i-$th square root digit: $y_i \in D_S$
Y_i	partially developed square root with i digits ($Y_0 = 0$ and $Y_i = Y_{i-1} + y_i B^{-i}$)
Y, Y_n	square root value
δ	number of fractional bits of X_{i-1}^* which are inspected for digit selection
f	number of bits to the left of the point of X_{i-1}^* which are inspected for digit selection
$\widehat{X}_{i-1}^*$	truncated value of X_{i-1}^* to the $\delta-$th fractional bit in non redundant representation
r	$= f + \delta$; total number of bits of $\widehat{X}_{i-1}^*$
ε	$= X_{i-1}^* - \widehat{X}_{i-1}^*$; $\varepsilon \in (\varepsilon_L, \varepsilon_H)$, with $\varepsilon_L < \varepsilon_H$
$\Delta\varepsilon$	$= \varepsilon_H - \varepsilon_L$
k	number of fractional bits of Y_i which are inspected for digit selection
$\widehat{Y}$	truncated value of Y_i to the $k-$th fractional bit

relationships between all these parameters will allow the designer, once the radix and the digit set have been selected, to determine the representation of the operand and the result of the adder, and then obtain the minimum possible sizes for CLA, LT and DST, as well as the contents of LT and DST.

3 Square rooting algorithm

3.1 Region of convergence

The region of convergence of the algorithm is determined by considering the transformations of the shifted partial remainder when the square root algorithm implies a selection respectively of the largest and smallest digits of the digit set. The details of the operations which lead to the final result can be found in [3]. Specific region of convergences (and corresponding selection rules) have been computed in the case of radix 2 square rooting, and can be found in [11] and in [12].

It can be demonstrated that the largest region of convergence of the square rooting algorithm for $i > 1$ is expressed by relation (3).

$$-2\eta Y_{i-1} + B\eta^2 B^{-i} \leq X^*_{i-1} \leq +2\eta Y_{i-1} + B\eta^2 B^{-i} \quad (3)$$

where

$$\eta = \frac{s}{B-1}$$

is also called index of redundancy and it is always $1/2 < \eta \leq 1$. It can be observed that for $i = 0$ in general, X does not belong to the region of convergence. Moreover, it can be demonstrated [3] that, in order to have a correct square root extraction, Theorem 1 must hold.

Theorem 1 *The first "digit" of the square root value y_1, satisfies the relation $(B/2) \leq y_1 \leq B$. If $\eta = 1$ it is possible to reduce the range to $(B/2) < y_1 < B$.*

The importance of the region of convergence is strictly related to the determination of the minimal number of bits f of each partial remainder $\widehat{X}^*_{i-1}$ which lie *before the point* and which need to be examined together with the δ fractional bits in order to perform the square rooting process.

Let us recall the expression of the region of convergence. For all the possible values of X^*_{i-1} which belong to the region of convergence, we have to derive the domain of the corresponding *truncated* results $\widehat{X}^*_{i-1}$. Since $\widehat{X}^*_{i-1} = X^*_{i-1} - \varepsilon$ with $\varepsilon_L \leq \varepsilon \leq \varepsilon_H$, and iterations occur for values of $i \geq k/b + 1$, which however implies $Y_{i-1} \leq (1 - BB^{-i})$, all the values X^*_{i-1} which fall within the region of convergence, certainly originate truncations $\widehat{X}^*_{i-1}$ which satisfy the relation (4) below.

$$-2\eta - \varepsilon_H < \widehat{X}^*_{i-1} < 2\eta - \varepsilon_L \quad (4)$$

The value f must allow the representation in two's complement of the largest absolute value of the two bounds of the relation (4).

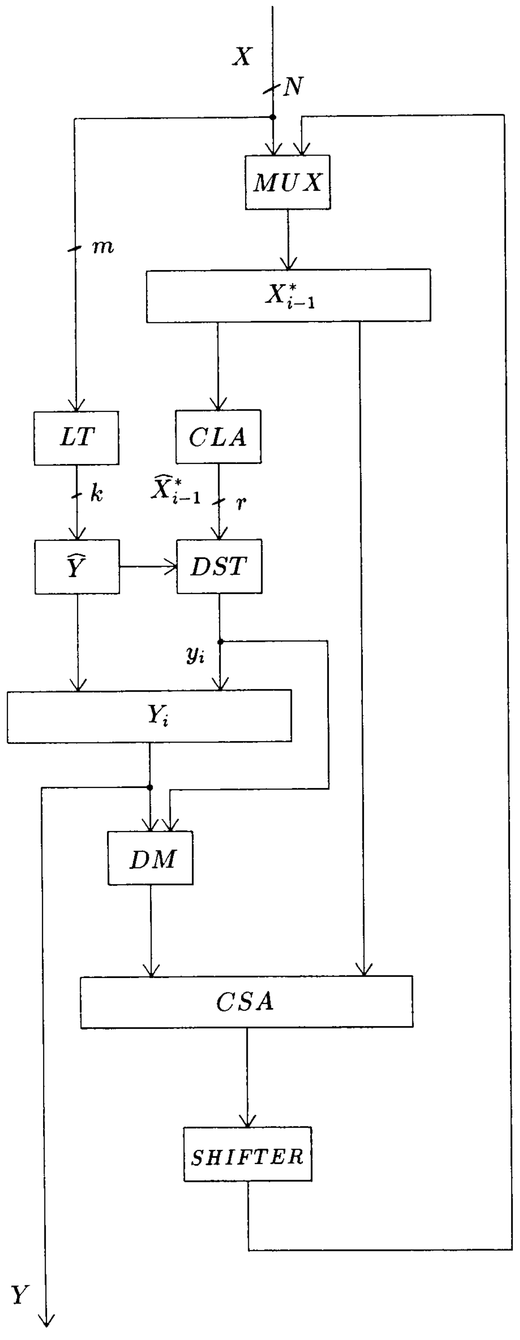

Figure 1: The layout of the square root extractor

3.2 Intervals for digit selection

The bounds of the intervals for digit selection can be computed with a similar approach to the one followed for the determination of the region of convergence. As pointed out in [4], (although this paper refers more specifically to an application of DeLugish algorithms), the intervals for digit selection are derived by requiring that the new shifted partial remainder still belongs to the region of convergence of relation (3). As described in the detail in [3], the interval which causes the selection of the digit $y_i = j$ for steps $i > 1$ results as

$$2\phi_j Y_{i-1} + B\phi_j^2 B^{-i} \leq X_{i-1}^* \leq 2\theta_j Y_{i-1} + B\theta_j^2 B^{-i} \quad (5)$$

where

$$\phi_j = \frac{-\eta + j}{B} \quad \text{and} \quad \theta_j = \frac{\eta + j}{B} \quad (6)$$

For the first steps it is necessary to make use of a lookup table which implements the relation (29) which will be introduced in section 5.

3.3 Remarks

From these results we observe that the project of a higher radix square root extractor involves the design of two sub-units: a lookup and a selection sub-unit. The task of the lookup unit is to carry out the initialization phase with the determination of some digits of the final result (i.e. a starting value $\widehat{Y}$), by inspecting some bits of the first radicand X. The aim of the lookup table is to produce a convenient transformation of the first radicand into a shifted radicand (for a defined starting value of i) which lies inside the region of convergence. Moreover, the determination of the "first" k fractional bits of the result, which are required by the iterative process of the second phase, can be accomplished referring again to the lookup table. Conversely, the task of the selection sub-unit is to perform the step-by-step root extraction. The layout of the complete square root extractor, (sketched in Fig. 1), has already been extensively described in section 2.

In this paper we will focus our attention mostly on the design of the selection sub-unit and, in particular, on the design of its selection table. In fact, as the extensive computations can show [3], the project of the lookup table can be, for most parts, derived by following a similar approach to the one which is shown below.

In section 4 the elimination of the dependence of the bounds of the relation (5) on i will be examined, followed by an investigation of the evolutions of the same intervals when the values Y_{i-1} and X_{i-1}^* are inspected only over a reduced set of bits. The goal will be to find the minimal number of bits of Y_{i-1} and of X_{i-1}^* which have to be inspected so as to ensure correct digit selection.

4 Selection table

As has been outlined in a previous section, the square rooting process starts with a convenient lookup in a table which returns a result $\widehat{Y}$ in non redundant form over k fractional bits. In our assumptions this value is stored in a register and *left unchanged* even if successive iterations would allow it to differ from the first bits of the square root Y under computation. Since the value $\widehat{Y}$ will be used as an entry to the digit selection table, this assumption can imply simpler hardware circuits. From Theorem 1 it is clear that $1/2 \leq \widehat{Y} \leq 1$ for $\eta < 1$, a range which can be reduced to $1/2 < \widehat{Y} < 1$ for $\eta = 1$. The square rooting process then starts from $i = (k/b) + 1$.

From the definition of $\widehat{Y}$ it is clear that for $n \to \infty$

$$Y_n \in \left[\max\{\widehat{Y} - \eta 2^{-k}, \frac{1}{2}\}, \min\{\widehat{Y} + \eta 2^{-k}, 1\}\right)$$

because $\eta 2^{-k}$ is the maximum value which can be represented with an infinite sequence of bits starting from the $(k+1)$-th fractional position with the digit set D_s.

Besides,

$$\begin{aligned} Y_{i-1} \in \; & \left[\max\{\widehat{Y} - (\eta 2^{-k} - \eta 2^{-(i-1)b}), \tfrac{1}{2}\},\right. \\ & \left.\min\{\widehat{Y} + (\eta 2^{-k} - \eta 2^{-(i-1)b}), 1\}\right] \end{aligned} \quad (7)$$

To perform the truncation of Y_{i-1} into the value $\widehat{Y}$ correctly it is necessary to remember the assumption $k > 0$. Since the non-linear operators max and min appear in relation (7), it is necessary to split the analysis into three sections, namely $1/2 < \widehat{Y} < 1, \widehat{Y} = 1/2$ and $\widehat{Y} = 1$. Moreover, for each section separate investigations have to be carried out for the digits $j < 0, j = 0$ and $j > 0$.

4.1 Analysis of the case $1/2 < \widehat{Y} < 1$

4.1.1 Selection rules

Let us consider the negative digits, that is $j < 0$. From their definition in equations (6) it follows $\phi_j < 0, \theta_j \leq 0$. The reduction $Y_{i-1} \to \widehat{Y}$ changes the lower bound of the interval (when $j < 0$), i.e. relation (5), into a value which does not depend on i.

$$\begin{aligned} 2\phi_j Y_{i-1} + B\phi_j^2 B^{-i} & \leq 2\phi_j[\widehat{Y} - \eta 2^{-k} + BB^{-i}(\eta + \frac{\phi_j}{2})] \\ & \leq 2\phi_j[\widehat{Y} - \eta 2^{-k}] \quad (8) \end{aligned}$$

A similar reasoning leads to the determination of the new upper bound and the generic new interval which is valid for negative digit selection and which is independent of the value of i.

$$2\phi_j[\widehat{Y} - \eta 2^{-k}] \leq X_{i-1}^* \leq 2\theta_j[\widehat{Y} + \eta 2^{-k}] \quad (9)$$

With simple passages it is possible to determine the expressions of the new intervals for $j = 0$ and $j > 0$. The results are similar to the one expressed in relation (9). We now apply truncation on the remainder X_{i-1}^*. In other words, we deal only with $\widehat{X}_{i-1}^*$ which represents the value X_{i-1}^* only from its most significant bit to the $2^{-\delta}$ weight. Then,

$$\widehat{X}_{i-1}^* \in [X_{i-1}^* - \varepsilon_L, X_{i-1}^* - \varepsilon_H]$$

The selection rules of digit $y_i = j$ result

$$j < 0;$$
$$2\phi_j[\widehat{Y} - \eta 2^{-k}] - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_j[\widehat{Y} + \eta 2^{-k}] - \varepsilon_H \quad (10)$$
$$j = 0;$$
$$2\phi_0[\widehat{Y} - \eta 2^{-k}] - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_0[\widehat{Y} - \eta 2^{-k}] - \varepsilon_H \quad (11)$$
$$j > 0;$$
$$2\phi_j[\widehat{Y} + \eta 2^{-k}] - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_j[\widehat{Y} - \eta 2^{-k}] - \varepsilon_H \quad (12)$$

The relations (10), (11), (12) express the widest conditions which are still conservative with respect to the relation (5).

4.1.2 Discretization

The values of the bounds are expressed in N bits just as the radicand X is. Therefore, digit selection by means of the relations (10), (11), (12) would involve a comparison between numbers which are N bits long. Without altering the validity of the algorithm it is possible to limit the tests required by relations (10), (11), (12) to comparisons between values $r = f + \delta$ bits long. In this section the discretization process of the bounds of the intervals will be analyzed, in order to obtain a new set of intervals which completely cover the range of the X values, leaving no gaps.

Let us consider the intervals corresponding to two consecutive digits $j-1$ and j. In order to ensure a correct square rooting process it is necessary for the corresponding *discretized* intervals to be at least consecutive with no *gaps* between one interval and the next.

Let us denote with

$$low_{j-1} \leq \widehat{X}^*_{i-1} \leq up_{j-1} \quad \text{and} \quad low_j \leq \widehat{X}^*_{i-1} \leq up_j$$

the two selection intervals (in analytical form) corresponding to the digits $j-1$ and j. It is known that the discretization process does not introduce gaps between the upper bound of one interval and the lower bound of the other if

$$up_{j-1} - low_j \geq 0 \quad (13)$$

A correct result can be obtained from the discretization process if the relation (13) is satisfied for all the intervals in the original form. However, the relation (13) does not always provide the best results; even if the relation (13) is not satisfied, there could be some cases where the specific continuous values of the bounds after discretization, do not insert any *gap* between one interval and the next. We define as *Truncation Factor*, (and we represent it with the symbol L_f), a multiplicative coefficient of the discretization step D which allows the relation (13) to be generalized.

$$up_{j-1} - low_j \geq -L_f \cdot D \quad (14)$$

When $L_f = 0$ relation (14) becomes relation (13). Moreover, it is easy to prove that the condition $L_f > 1$ introduces gaps in the discretization process. Therefore, using the relation (14) with $L_f = 1$, it is possible to obtain the lower limit on the number of bits (k, δ) which have to be inspected. Actually, it is not guaranteed that this bound can be reached, since the intervention of $L_f = 1$ depends exclusively on the specific values of η and B.

We perform our computations imposing the conditions expressed in relation (13), leaving the discussion of how to use the relation (14) with $D = 2^{-\delta}$ to section 4.4.

Before applying relation (13) between the intervals of the two consecutive digits $\{j-1, j\}$, it is necessary to split the analysis into two cases: $j \leq 0$ and $j \geq 1$.

If $j \leq 0$ then $\theta_{j-1} \leq 0, \phi_j < 0$. The imposition of relation (13) on the expression (10) with $j-1 = -s$ identifies the *critical* situation which is represented by expression (15).

$$\left(\frac{2\eta - 1}{B}\right)\widehat{Y} \geq \frac{\Delta\varepsilon}{2} + \left[\frac{2\eta(B-1) - 1}{B}\right]\eta 2^{-k} \quad (15)$$

A similar approach is followed in the alternative case $j \geq 1$. The results coincide with the expression (15). Then, expression (15) is the basis of a formula which correlates the values k and δ (which is hidden inside the term $\Delta\varepsilon$). It is necessary to remember here that in this section we are working under the condition $1/2 < \widehat{Y} < 1$. Therefore, by substituting into expression (15) the value $min\{\widehat{Y}\} = 1/2 + 2^{-k}$, we obtain the following result

$$\left[\frac{2\eta - 1}{B}\right]\frac{1}{2} \geq \frac{\Delta\varepsilon}{2} + \left[\frac{2\eta^2(B-1) - 3\eta + 1}{B}\right]2^{-k} \quad (16)$$

which yields the condition (valid if the argument of the logarithm is positive)

$$k \geq \left\lceil \log_2\left(\frac{2\eta^2(B-1) - 3\eta + 1}{\frac{1}{2}(2\eta - 1) - \frac{B}{2}\Delta\varepsilon}\right)\right\rceil \quad (17)$$

4.2 Analysis of the case $\widehat{Y} = 1/2$

4.2.1 Selection rules

A preliminary consideration is necessary. Since we are working with $1/4 \leq X < 1$, then would be $1/2 \leq Y < 1$. Therefore, for if X^*_{i-1} is negative, the partial radicand must be $Y_{i-1} > 1/2$. If not, a negative digit would be introduced in the computation of Y which would be impossible to recover. This would lead to a final result of less than 1/2, which is contrary to the hypothesis. Hence, (remembering that in this section $\widehat{Y} = 1/2$), $X^*_{i-1} < 0$ implies $Y_{i-1} \geq \widehat{Y} + 2^{-(i-1)b}$ This, after some passages, yields the complete summary of the selection rules for digit $y_i = j$, which express the widest conservative conditions with respect to relation (5).

$$j < 0; \quad 2\phi_j\widehat{Y} - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_j[\widehat{Y} + \eta 2^{-k}] - \varepsilon_H \quad (18)$$
$$j = 0; \quad 2\phi_0\widehat{Y} - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_0\widehat{Y} - \varepsilon_H \quad (19)$$
$$j > 0; \quad 2\phi_j[\widehat{Y} + \eta 2^{-k}] - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_j\widehat{Y} - \varepsilon_H \quad (20)$$

4.2.2 Discretization

Again, by following the same approach as in section 4.1.2, and remembering that (in this paragraph) $\widehat{Y} = 1/2$, we get to expression (21).

$$\left[\frac{2\eta - 1}{B}\right]\frac{1}{2} \geq \frac{\Delta\varepsilon}{2} + \left[\frac{\eta^2(B-2)}{B}\right]2^{-k} \quad (21)$$

Finally, we obtain the condition (valid when the argument of the logarithm is positive)

$$k \geq \left\lceil \log_2 \left(\frac{\eta^2(B-2)}{\frac{1}{2}(2\eta - 1) - \frac{B}{2}\Delta\varepsilon} \right) \right\rceil \tag{22}$$

4.3 Analysis of the case $\widehat{Y} = 1$

Although, the case $\widehat{Y} = 1$ is not involved in the equations for the minimization of k and δ since the *critical* situations hold for values of $\widehat{Y}$ near 1/2 (see the expressions (15) and (21)), it is necessary to find an analytical expression for the intervals so that the solution can be completed. Since the computations involved are similar to the ones illustrated in the previous sections, we only mention the results. The complete summary of the selection rules for digit $y_i = j$, which express the widest conservative conditions with respect to relation (5), results

$$j < 0;$$
$$2\phi_j[\widehat{Y} - \eta 2^{-k}] - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_j \widehat{Y} - \varepsilon_H \tag{23}$$
$$j = 0;$$
$$2\phi_0[\widehat{Y} - \eta 2^{-k}] - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_0[\widehat{Y} - \eta 2^{-k}] - \varepsilon_H \tag{24}$$
$$j > 0;$$
$$2\phi_j \widehat{Y} - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_j[\widehat{Y} - \eta 2^{-k}] - \varepsilon_H \tag{25}$$

4.4 Remarks

In the previous sections we have presented a set of formulae which define certain conditions which can be used to determine the values of k and δ. The task of this section is to discuss the results, in order first to determine the condition which is the most restrictive and second to identify the pair of values (k, δ) of interest to the designer. A comparison between the expressions (16) and (21) shows that the most restrictive condition, namely the expression (16), occurs when $\widehat{Y} = 1/2 + 2^{-k}$ which is in every case a common value to both minimal $\widehat{Y}$ ranges when $\eta < 1$ and when $\eta = 1$. Actually, since $B > 2$ it is possible to adopt expression (17). Actually, the relations (10), (11), (12), (18), (19), (20), (23), (24), (25) can be joined together in a more compact form.

$$\begin{array}{lll} j < 0 & \rightarrow & 2\phi_j A_k - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_j B_k - \varepsilon_H \\ j = 0 & \rightarrow & 2\phi_0 A_k - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_0 A_k - \varepsilon_H \\ j > 0 & \rightarrow & 2\phi_j B_k - \varepsilon_L \leq \widehat{X}^*_{i-1} \leq 2\theta_j A_k - \varepsilon_H \end{array} \tag{26}$$

where

$$A_k = [\max(\widehat{Y} - \eta 2^{-k}, \frac{1}{2})] \quad \text{and} \quad B_k = [\min(\widehat{Y} + \eta 2^{-k}, 1)]$$

Because the condition (17) is derived from relation (13), it expresses a sufficient relation which a pair of values (k, δ) must satisfy in order to guarantee a correct square rooting. However, the choice of the effective values k, δ which are necessary for the physical implementation of the square root unit, is strictly related to a function of cost F_C, i.e. it depends on technological constraints. F_C is deduced by observing that k and δ affect in different ways the size of the table of the selection rules, the size of the addition circuits, the dimension of the lookup table for the first phase, the number of data paths and so on.

The function of cost F_C cannot be simply determined without any information concerning the design details and the technological constraints related to the implementation. The results presented so far may be used for the main purposes to identify constraints on the values (k, δ) which define the feasibility region representing the domain where the designer has to search for the best solution, and to offer methodologies with which to determine the contents of both the lookup table (if any) used in the initial phase, and the digit selection table. In order to determine the feasibility region, it is necessary to take into account the following considerations:

1. the values k and δ have to be integers;
2. the results which are provided by expression (13) are related to a sufficient condition; $L_f > 0$ could improve them.

Actually, from the second consideration, it follows that it is possible to identify two types of regions. The first is a ***sufficient*** region and obtained by considering the sufficient condition (13) (where it is implicitly assumed $L_f = 0$), which leads to (16) and (17). Within this region it is guaranteed that for all the pairs (k, δ) belonging to this region it is possible to obtain valid digit selection tables. It represents the largest domain for which this is true. The other region is obtained from (14) with $L_f = 1$. By applying the same methodology used to get relations (16) and (17) from (13), it is possible to derive from (14) formulae (27) and (28) which define this *extended* region.

$$\left[\frac{2\eta - 1}{B}\right]\frac{1}{2} \geq \frac{(\Delta\varepsilon - 2^{-\delta})}{2} + \left[\frac{2\eta^2(B-1) - 3\eta + 1}{B}\right]2^{-k} \tag{27}$$

$$k \geq \left\lceil \log_2 \left(\frac{2\eta^2(B-1) - 3\eta + 1}{\frac{1}{2}(2\eta - 1) - \frac{B}{2}(\Delta\varepsilon - 2^{-\delta})} \right) \right\rceil \tag{28}$$

The *extended* region includes the ***sufficient*** region, since if (16) holds for a pair (k, δ), then (27) also holds, but the converse is not necessarily true. The ***complementary*** region of the ***sufficient*** with respect to the ***extended*** region, includes pairs (k, δ) where (14) holds only with $0 < L_f \leq 1$. As $L_f > 0$ violates the sufficient condition expressed by (13), it is not guaranteed that it is possible to obtain a valid digit selection table for all the pairs (k, δ) of the ***complementary*** region. Since L_f cannot be controlled by the designer, but is a consequence of the discretization process, the ***complementary*** region could be analyzed by checking the validity of the selection tables corresponding to all the pairs (k, δ) of interest to the designer. A general formulation of the problem in terms of all the possible values (B, η) can be approached only by introducing the coefficient L_f. This is because it is not possible to study the specific numerical properties of the single coefficients in such a general case, as has been possible in the particular cases presented in [2], [5], [6], [11] and [13]. At this point, it is worth noting that Theorem 1 coupled with the presence of the register for $\widehat{Y}$, avoids, when $\eta = 1$, the need of having the entries $\widehat{Y} = 1/2$ and $\widehat{Y} = 1$ in the selection table.

5 Lookup table

The design of the lookup table can be, for some parts (see [3]), derived from the analysis which we have provided in the previous sections. However, it can be observed that a lookup table is necessary only for $k > 0$ if $\eta < 1$ and for $k > 1$ if $\eta = 1$. The *key formula*, actually relation (29), which rests on the basis of the determination of the lookup table, can be derived directly from the expression governing the region of convergence.

$$(\widehat{Y} - \eta 2^{-k})^2 - \varepsilon_{XL} \leq \widehat{X} \leq (\widehat{Y} + \eta 2^{-k})^2 - \varepsilon_{XH} \qquad (29)$$

where m is the number of fractional bits of X which enter the lookup table and $\widehat{X}$ denotes the truncation of X to its $m-th$ fractional bit and where $\varepsilon_X = X - \widehat{X}$, with $\varepsilon_X \in (\varepsilon_{XL}, \varepsilon_{XH})$ and $\varepsilon_{XL} < \varepsilon_{XH}$, and $\Delta\varepsilon_X = \varepsilon_{XH} - \varepsilon_{XL}$ with $\Delta\varepsilon_X \geq 2^{-m}$ As it has been illustrated, the effective value of the number of fractional bits of X has to be computed by determining the minimal m which provides a correct lookup table. This is true because the function of cost of the lookup table depends only on m. Again, in most cases, the number of tests to be performed is very small, since it can be demonstrated [3] that for $\eta < 1$ it is $k \leq m \leq m_{max}$, and for $\eta = 1$ it is $(k-1) \leq m \leq m_{max}$ with

$$m_{max} = k + \left\lceil \log_2\left(\frac{1}{(2\eta - 1)(1 + 2^{-k})}\right) \right\rceil \qquad (30)$$

6 A case of study: radix 4 square rooting

Although two possibilities exist for the radix 4 square root extraction, namely $\eta = 1$ and $\eta = 2/3$, only the latter will be considered since it involves simpler hardware circuits. As in [11], we will consider X^*_{i-1} represented in carry-save form. In other words, let $\Delta\varepsilon = 2 \cdot 2^{-\delta}, B = 4, \eta = 2/3$, and $\varepsilon_L = 0$. In the *complementary* region we consider the following pairs which correspond to small values of (k, δ): $(k = 4, \delta = 5), (k = 5, \delta = 5), (k = 6, \delta = 4), (k = 7, \delta = 4), (k = 8, \delta = 4), (k = 9, \delta = 4), (k = 10, \delta = 4)$ The computation of the corresponding table contents shows that only the pair $(k = 5, \delta = 5)$ provides valid results.

The intervals for digit selection in the case $(k = 5, \delta = 5)$ are reported in detail for each $\widehat{Y}$ in Table 2 and correspond to $L_f = 1/6$. It is worth noting that the lower and the upper bounds of the intervals, which correspond to the most negative and most positive digits respectively, are in effect the bounds of the region of convergence, and have to be computed by means of relation (3), or in a general form by relation (4).

In order to determine the $i-th$ digit y_i, it is necessary to enter Table 2 with the two values $\widehat{Y} \cdot 2^k$ and $\widehat{X}^*_{i-1} \cdot 2^\delta$. The value $\widehat{Y} \cdot 2^k$ determines the row of the Table containing the intervals for digit selection, and the value of $\widehat{X}^*_{i-1} \cdot 2^\delta$, by means of such intervals, selects the digit y_i. For example,

Table 2: Radix 4 : the intervals for digit selection for $k = 5, \delta = 5$

$\widehat{Y} \cdot 2^k$	digit y_i							
	-2	-1		0		1		2
	hi.	lo.	hi.	lo.	hi.	lo.	hi.	lo.
16	-14	-13	-5	-5	3	3	11	12
17	-14	-13	-5	-5	3	3	11	12
18	-15	-14	-6	-5	3	4	12	13
19	-16	-15	-6	-6	4	4	13	14
20	-16	-16	-6	-6	4	4	14	14
21	-17	-16	-6	-6	4	4	14	15
22	-18	-17	-6	-7	5	4	15	16
23	-18	-18	-6	-7	5	4	16	16
24	-19	-19	-7	-7	5	5	17	17
25	-20	-20	-7	-8	6	5	18	18
26	-20	-21	-7	-8	6	5	19	18
27	-21	-21	-7	-8	6	5	19	19
28	-22	-22	-7	-9	7	5	20	20
29	-22	-23	-7	-9	7	5	21	20
30	-23	-24	-8	-9	7	6	22	21
31	-24	-25	-8	-10	8	6	23	22
32	-24	-26	-8	-10	8	6	24	22

Table 3: Radix 4: the lookup table for $k = 5, \delta = 5$

intervals					$\widehat{Y} \cdot 2^k$
32	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	33	16
34	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	38	17
38	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	42	18
43	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	47	19
47	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	52	20
52	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	57	21
57	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	63	22
63	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	69	23
69	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	75	24
75	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	81	25
81	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	87	26
87	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	94	27
94	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	101	28
101	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	109	29
108	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	116	30
116	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	124	31
123	$\leq$	$\widehat{X} \cdot 2^m$	$\leq$	127	32

the digit selected by observing Table 2 when $\widehat{Y} \cdot 2^5 = 18$ and $\widehat{X}^*_{i-1} \cdot 2^5 = -4$ is $y_i = 0$, while when $\widehat{X}^*_{i-1} \cdot 2^5 = 7$ is $y_i = 1$.

After the first lookup, the second phase starts with $i = k/b + 1 = 7/2$. From relation (4) the number f of integer bits of $\widehat{X}^*_{i-1}$, can be computed. Since it is necessary to represent in two's complement quantities in the range $(-67/48, 4/3)$, $f = 2$ bits are sufficient. Actually, the global inspection of $r = f + \delta = 7$ bits of $\widehat{X}^*_{i-1}$ is necessary and sufficient to perform the correct square rooting.

This is an improvement with respect to the 9 or 10 bits of [15] and the 8 bits of [8]. With reference to the relations in section 5, it can be foreseen that, for $\Delta\varepsilon_X = 2^{-m}$ and $\varepsilon_{XL} = 0$, which correspond to a radicand X given in non-redundant form, it follows that $5 \le m \le 7$. Actually, the computation of the lookup table, which is reported in Table 3, yields a first global inspection of $m = 7$ bits.

7 Conclusions

This paper has presented a set of formulae delimiting the feasible space of all the non-restoring square root algorithms, by correlating radix, digit set and internal representation of the partial remainder. Moreover, it has been shown that it is possible to develop a methodology for computing the contents of the lookup tables required for digit selection and for obtaining the initial radicand value. The definition of the feasibility space has been used to minimize some important parameters, such as the number of bits of the partial remainder to be inspected to determine the new result digit and the number of radicand bits to be inspected to generate the first root value.

The methodology presented has been applied to the practical case of radix 4, digit set $[-2, +2]$ and partial remainder represented as produced by a carry-save adder; the algorithm obtained reduces the bits to be inspected for digit selection from 8 [8] or 9 [15] to 7, thus improving both speed and complexity of the whole unit.

References

[1] D. E. Atkins, "Higher-Radix Division Using Estimates of the Divisor and Partial Remainders," IEEE Trans. Comput., Vol.C-17, pp.925-934, October 1968.

[2] L. B. Bushard, "A Minimum Table Size Result for Higher Radix Nonrestoring Division," IEEE Trans. Comput., Vol.C-32, pp.521-526, June 1983.

[3] L. Ciminiera and P. Montuschi, "Higher Radix Square Rooting," Internal Report, Politecnico di Torino, Dipartimento di Automatica e Informatica, I.R. DAI/ARC 4-87, December 1987.

[4] M. D. Ercegovac, "Radix-16 Evaluation of Certain Elementary Functions," IEEE Trans. Comput., Vol.C-22, pp.561-566, June 1973.

[5] M. D. Ercegovac, "An On-Line Square Rooting Algorithm," Proc. 4th IEEE Symposium on Computer Arithmetic, Santa Monica, CA, pp.183-189, October 1978.

[6] M. D. Ercegovac and T. Lang, "On-Line Computation of Rotation Factors," Proc. 8th IEEE Symposium on Computer Arithmetic, Como, Italy, pp.196-203, May 1987.

[7] M. D. Ercegovac and T. Lang, "On-the-Fly Conversion of Redundant into Conventional Representations," IEEE Trans. Comput., Vol.C-36, pp.895-897, July 1987.

[8] J. Fandrianto, "Algorithm for High Speed Shared Radix 4 Division and Radix 4 Square-Root," Proc. 8th IEEE Symposium on Computer Arithmetic, Como, Italy, pp.73-79, May 1987.

[9] M. Homewood, D. May, D. Shepherd and R. Shepherd, "The IMS T800 Transputer," IEEE Micro, Vol.7, No.5, October 87, pp.10-26.

[10] "IEEE Standard for Binary Floating-Point Arithmetic" IEEE Standard 754, 1985 IEEE Computer Society.

[11] S. Majerski, "Square-Rooting Algorithms for High-Speed Digital Circuits," IEEE Trans. Comput., Vol.C-34, pp.724-733, August 1985.

[12] G. Metze, "Minimal Square Rooting," IEEE Trans. Electron. Comput., Vol.EC-14, pp.181-185, April 1965.

[13] V. G. Oklobdzija and M. D. Ercegovac, "An On-Line Square Root Algorithm," IEEE Trans. Comput., Vol.C-31, pp.70-75, January 1982.

[14] J. E. Robertson, "A New Class of Digital Division Methods," IRE Trans. Electron. Comput., Vol.EC-7, pp.218-222, September 1958.

[15] J. H. P. Zurawski and J. B. Gosling, "Design of a High-Speed Square Root Multiply and Divide Unit," IEEE Trans. Comput., Vol.C-36, pp.13-23, January 1987.