Leading Guard Digits in Finite-Precision Redundant Representations

Jean-Michel Muller, Peter Kornerup

Leading Guard Digits in Finite Precision Redundant Representations

Peter Kornerup, *Member*, *IEEE*, and Jean-Michel Muller, *Senior Member*, *IEEE*

Abstract—Redundant number representations are generally used to allow constant time additions, based on the fact that only bounded carry-ripples take place. But, carries may ripple out into positions which may not be needed to represent the final value of the result and, thus, a certain amount of leading guard digits are needed to correctly determine the result. Also, when cancellation during subtractions occurs, there may be nonzero digits in positions not needed to represent the result of the calculation. It is shown here that, for normal redundant digit sets with radix greater than two, a single guard digit is sufficient to determine the value of such an arbitrary length prefix of leading nonzero digits. This is also the case for the unsigned carry-save representation, whereas two guard digits are sufficient, and may be necessary, for additions in the binary signed-digit and 2's complement carry-save representations. Thus, only the guard digits need to be retained during sequences of additions and subtractions. At suitable points, the guard digits may then be converted into a single digit, representing the complete prefix.

Index Terms—Redundant representations, leading guard digits, multioperand additions, pseudo overflows.

1 Introduction and Notation

In hardware realizations of many calculations, a redundant representation of operands is advantageous as it allows addition and subtraction to be performed in constant time, independent of the word size. Also, an argument to a multiplier may be delivered in a redundant representation as recoding into a higher radix can take place directly from such a representation. Similarly, it is often not necessary to convert the result of a multiplication or division to nonredundant form, as it may have to participate in further computations. In general, it is preferable to leave intermediate results in redundant form until they eventually must be converted to a nonredundant representation as the cost of such conversions is at least logarithmic in the word size. Observe that, for a redundant representation, there are no simple (constant time) overflow checks, so a computation must be planned such that there is no overflow in the final result. Thus, it is necessary to design circuits with an appropriate number of digit positions to assure that there is room for the worst-case situation.

But, redundant representations may have "nonsignificant" prefix-strings of digits, e.g., binary signed-digit representations may have prefix strings of the form $\bar{1}11\cdots1 \sim \bar{1}$. In many applications (SRT division, elementary functions with redundant remainder), we need information on the partial remainder (either its approximate value, its sign, or knowing it is small) from a small window of digits, without converting it to nonredundant form. Very often a bound on the result of a computation is known and there is no need to use more leading digit positions (guard digits) than absolutely necessary to identify the value of such a prefix string and, thus, the value of the number represented. As an example, in the SRT division algorithm as introduced by Atkins in [1], employing remainder updating in a redundant representation, the digit selection is based on some leading digits of the remainder whose value is known to be bounded, but obtained by an effective subtraction with cancellation. In an implementation, a sufficient amount of such guard digits must be maintained for a correct digit selection to be possible. In a radix 2^k SRT divider, due to cancellation at least k leading digits of value zero are generated, but they are not necessarily represented by zero-digits. Also, in "shift-and-add" type of elementary function evaluations based on redundant "remainders," there is a similar need for considering a minimal amount of leading guard digits, e.g., as in [4, Section 5.3].

Here, we analyze the minimal number of guard digits needed to be able to determine the value of a redundantly represented number for which a bound on its value is known. Given such a bound, together with the number of digits needed to represent such bounded values, it is possible to obtain a small bound on the value represented by any additional prefix string of digits present in the redundant representation. It is then shown that this prefix value can always be determined by a few of the least significant digits of the prefix string, these being the guard digits. It is shown that, for contiguous digit sets with digits of absolute value less than or equal to the base $\beta \geq 3$, a single guard digit is sufficient, which is also the case for the (unsigned) carry-save representation, but that two guard digits are sufficient and may be necessary for binary signed-digit and 2's complement carry-save representations.

- *P. Kornerup is with the Department of Mathematics and Computer Science, University of Southern Denmark, Campusvej 55, DK-5230 Odense, Denmark. E-mail: kornerup@imada.sdu.dk.*
- *J.-M. Muller is with the Laboratoire LIP/Arenaire, ENS Lyon, 46 Allee d'Italie, F-69364 Lyon Cedex 07, France. E-mail: Jean-Michel.Muller@ens-lyon.fr.*

Manuscript received 17 Mar. 2005; revised 4 Sept. 2005; accepted 21 Nov. 2005; published online 22 Mar. 2006.
For information on obtaining reprints of this article, please send e-mail to: tc@computer.org, and reference IEEECS Log Number TC-0081-0305.

The present work was initiated in continuation of an analysis of the digit selection process in SRT division as presented in [3]. Previous work includes [7], presenting digit-sequential algorithms for detecting "real overflow" and correcting for "nonreal" (pseudo) overflow in signed-digit number systems. While discussing the use of carry-save representations in signal processing, [6] considers the problem of converting prefix strings in 2's complement carry-save representations and provides simple constant time conversions and [8] similarly describes a necessary correction in the instance of adding two signed-digit numbers. We shall return to the two latter results.

Below, we shall briefly introduce our notation for radix polynomials and Section 2 will contain a fundamental lemma and results where a single guard digit turns out to be sufficient. Sections 3 and 4 deal with the binary borrow-save (signed-digit) representation and the more complex case of carry-save 2's complement, where two guard digits are needed. Section 4 finally discusses the results and presents the conclusions.

1.1 Notation

In this paper, we will use a somewhat formal notation of radix polynomials as representations of numbers, to be distinguishable from the values represented. Given an integer *radix* or *base* β, $|\beta| \geq 2$, a *radix* β *polynomial* is an expression of the form:

$$P = \sum_{i=\ell}^{m} d_i[\beta]^i, \tag{1}$$

where the *digits* d_i are integers, $d_i \in \mathbb{Z}$, belonging to some *digit set* D which is finite and such that $0 \in D$. The square brackets $[\,]$ around β are used to distinguish the radix polynomial P in (1) from its real value, which is denoted $\|P\|$, and can be expressed as:

$$\|P\| = \sum_{i=\ell}^{m} d_i\beta^i. \tag{2}$$

This allows us to discuss different representations of the same number or value. The representations (1) are assumed to have a finite number of terms, hence their values (2) are rational numbers. In the examples, we will also use string representations, using ordinary symbols for digit values, including the overbar to denote negative digit values.

Discussing radix representations, the base β will remain fixed and we will denote *the set of base* β *radix polynomials* over some digit set D by $\mathcal{P}[\beta, D]$, i.e.:

$$\mathcal{P}[\beta, D] = \left\{ \sum_{i=\ell}^{m} d_i[\beta]^i \;\mid\; d_i \in D,\ \ell \leq m \in \mathbb{Z} \right\}.$$

Also define the *fixed point radix polynomials* as a set of the form:

$$\mathcal{F}_{\ell,m}[\beta, D] = \left\{ P = \sum_{i=\ell}^{m} d_i[\beta]^i \mid P \in \mathcal{P}[\beta, D] \right\}$$

for some fixed, system defined, values of ℓ and m, $\ell \leq m$.

Similarly, we may define the *2's complement systems* by

$$\mathcal{F}^{2c}_{\ell,m}[2, \{-1, 0, 1\}] = \Big\{ P = \sum_{i=\ell}^{m+1} d_i[2]^i \mid d_i \geq 0 \text{ for } i = \ell, \ell+1, \cdots, m \text{ and } d_{m+1} = -d_m \Big\},$$

where the digit d_{m+1} is not represented in hardware realizations as its value can be derived from d_m. But, it is convenient to include it here for the analysis because it allows us to determine the value $\|P\|$ of a 2's complement polynomial P as for any other radix polynomial. Thus,

$$\|P\| = d_{m+1}2^{m+1} + \sum_{i=\ell}^{m} d_i 2^i = \sum_{i=\ell}^{m-1} d_i 2^i - d_m 2^m,$$

with range $-2^m \leq \|P\| \leq 2^m - 2^\ell$.

In the following, we shall, in particular, look at the redundant *2's complement carry-save polynomials* defined as the set

$$\mathcal{F}^{2c}_{\ell,m}[2, \{-2, -1, 0, 1, 2\}] = \Big\{ P = \sum_{i=\ell}^{m+1} d_i[2]^i \mid d_i \geq 0 \text{ for } i = \ell, \ell+1, \cdots, m \text{ and } d_{m+1} = -d_m \Big\},$$

for which the value of P belonging to $\mathcal{F}^{2c}_{\ell,m}[2, \{-2, -1, 0, 1, 2\}]$ may similarly be determined by

$$\|P\| = \sum_{i=\ell}^{m-1} d_i 2^i - d_m 2^m,$$

with range $-2^{m+1} \leq \|P\| \leq 2(2^m - 2^\ell)$.

2 Addition in Redundant Digit Sets

Addition in $\mathcal{F}_{\ell,m}[\beta, D]$ is not a closed operation in such a finite system, so precautions have to be taken if the result is not representable. Most often the basic integer addition operation of a computer does not notify the user in case of such an overflow situation, the system just discards digits (carry-outs) that cannot be represented and, thus, the computer implements a modular addition $(a + b) \bmod \beta^m$.

It is obvious that if the digit set D is nonredundant for the base β, then, for addition in $\mathcal{F}_{\ell,m}[\beta, D]$, any nonzero digit generated in position k with $k > m$ is a signal that the result is not representable. Thus, the carry generated in position m directly acts as an overflow signal. For addition in the nonredundant 2's complement system, the situation is different since position $m + 1$ in practice is not included in the representation. However, overflow can be detected when the carry-in to position m is different from the carry-out of that position.

If D is redundant for base β, the situation is more complicated. First, notice that if the value of the carry generated in the most significant position m is zero, then certainly the result is representable in $\mathcal{F}_{\ell,m}[\beta, D]$, but a nonzero carry does not necessarily imply that an overflow situation has arisen.

Example 1. Consider the following addition in $\mathcal{F}_{0,4}[2,\{-1,0,1\}]$:

$$\begin{array}{r} 0\ 1\ \bar{1}\ \bar{1}\ \bar{1} \\ 1\ 1\ 1\ 1\ 0 \\ \hline 1\ 0\ 0\ 0\ 0\ \bar{1} \end{array} \sim \quad 1\ 1\ 1\ 1\ 1$$

and note that the value of the result is actually representable in the system.

As the example indicates, a signaling of overflow must take at least $\mathcal{O}(\log(m-\ell))$ time, hence the advantage of constant time addition is lost if it is necessary to test for overflow following each addition. Fortunately, this may not be necessary in a composite computation when the result is known to satisfy a certain bound as long as a sufficient amount of *leading guard digits* are carried along in the computation to avoid loss of any significant information. Hence, with a little planning, overflow testing and, possibly, scaling of the result can be postponed until a conversion into a nonredundant representation may be needed anyway.

Since a number representation in a redundant digit set may have a prefix string of nonzero digits which can be reduced (e.g., $1\bar{1} \sim 01$ in signed-digit base 2), it may be necessary to be aware of the presence of such strings. It is well-known that up to two additional leading digits may be generated in such a redundant addition (one in the case of radix $\beta \geq 3$ with digit set $D = \{d \mid -\beta \leq -s \leq d \leq s \leq \beta\}$, where $2s \geq \beta + 1$, using Avizienis algorithm [2]). Fortunately, the value represented by such a prefix string is always bounded:

Lemma 1. *Let $P \in \mathcal{P}[\beta, D]$ with D be a digit set of the form $D = \{d \mid -\beta \leq r \leq d \leq s \leq \beta\}$, $s - r + 1 = \beta + \rho$, where the redundancy index $1 \leq \rho \leq \beta + 1$, and such that the value of P is representable in $\mathcal{F}_{\ell,m}[\beta, D]$:*

$$r\frac{\beta^{m+1} - \beta^{\ell}}{\beta - 1} \leq \|P\| \leq s\frac{\beta^{m+1} - \beta^{\ell}}{\beta - 1}.$$

Then, with Q and R defined by $P = Q[\beta]^{m+1} + R$, the absolute value of Q satisfies:

$$|\|Q\|| < 1 + \frac{\rho}{\beta - 1}$$

and, thus, if $\rho \leq \beta - 1$, it follows that $|\|Q\|| \leq 1$ and, for $\beta \geq 3$, the bound is $|\|Q\|| \leq 2$ when $\rho \in \{\beta, \beta + 1\}$.

Proof. From the definition of Q, we have $\|P\| = \|Q\|\beta^{m+1} + \|R\|$, hence

$$\begin{aligned} r\frac{\beta^{m+1} - \beta^{\ell}}{\beta - 1} - \|R\| &\leq \|Q\|\beta^{m+1} \\ &\leq s\frac{\beta^{m+1} - \beta^{\ell}}{\beta - 1} - \|R\|. \end{aligned}$$

But, $R \in \mathcal{F}_{\ell,m}[\beta, D]$ implies $r\frac{\beta^{m+1}-\beta^{\ell}}{\beta-1} - \|R\| \leq 0$ and $s\frac{\beta^{m+1}-\beta^{\ell}}{\beta-1} - \|R\| \geq 0$, then

$$\begin{aligned} |\|Q\|\beta^{m+1}| &\leq (s - r)\frac{\beta^{m+1} - \beta^{\ell}}{\beta - 1} \\ &= (\beta - 1 + \rho)\frac{\beta^{m+1} - \beta^{\ell}}{\beta - 1} \\ &< \left(1 + \frac{\rho}{\beta - 1}\right)\beta^{m+1}, \end{aligned}$$

from which the results follow, noting that $\|Q\|$ is integral. □

Observe that only in the extreme cases of $\rho = \beta$ or $\rho = \beta + 1$ for $\beta = 2$ is it possible that $|\|Q\|| \geq 2$, with a maximal bound of 3 obtained. This can be confirmed with $D = \{-2, -1, 0, 1, 2\}$, so $\rho = 3$, with $\|Q\| = 3$ in the example $11\bar{2}\bar{2}_2 = 6_{10} = 0022_2$. Later we shall discuss this digit set used in 2's complement carry-save representation. Note also that the situation where $\rho \leq \beta - 1$ includes cases with nonunique zero representation, i.e., where β (and/or $-\beta$) is a member of the digit set.

Theorem 2. *For $\beta \geq 3$, let $P = \sum_{i=\ell}^{m+k} d_i[\beta]^i \in \mathcal{F}_{\ell,m+k}[\beta, D]$, $k \geq 1$, be a polynomial whose value $\|P\|$ is representable in $\mathcal{F}_{\ell,m}[\beta, D]$, with digit set $D = \{d \mid -\beta \leq r \leq d \leq s \leq \beta\}$, $s - r + 1 = \beta + \rho$ with redundancy index $1 \leq \rho \leq \beta + 1$. Let Q be the prefix of P defined by $P = Q[\beta]^{m+1} + R$, then, for $1 \leq \rho \leq \beta - 1$, the value of $\|Q\|$ is bounded by $|\|Q\|| \leq 1$ and, for $\rho \in \{\beta, \beta + 1\}$, the bound is $|\|Q\|| \leq 2$. In both cases, $\|P\|$ can be uniquely determined by the relation $\|Q\| \equiv d_{m+1} \pmod{\beta}$, where d_{m+1} is the (guard) digit of P with weight β^{m+1}.*

Thus, conversion of $P \in \mathcal{F}_{\ell,m+k}[\beta, D]$, $k \geq 2$ into $P' \in \mathcal{F}_{\ell,m+1}[\beta, D]$ with $\|P'\| = \|P\|$ can be performed in constant time, exclusively based on converting the value of the single guard digit d_{m+1} into the value of Q and discarding all higher order terms.

Proof. Follows trivially from Lemma 1 since $\|Q\| \in \{-1, 0, 1\}$ implies that the only permissible values of d_{m+1} are $-\beta, -\beta + 1, -1, 0, 1, \beta - 1, \beta$, and, for $\|Q\| \in \{-2, 2\}$, the values are $-\beta + 2, -2, 2, \beta - 2$ since $\|Q\| \equiv d_{m+1} \pmod{\beta}$. □

The theorem shows that, for $\beta \geq 3$, one leading *guard digit* is sufficient to assure that the value of Q and, hence, $\|P\|$ can be identified. Note that, for $\beta = 2$, then 1 and -1 fall in the same residue class, hence the result does not apply. When $\|Q\| \equiv d_{m+1} \pmod{\beta}$ with $|\|Q\|| \leq 1$, then $\|Q\|$ is equal to the new digit d'_{m+1} of weight β^{m+1}, where d'_{m+1} can be expressed as

$$d'_{m+1} = \|Q\| = \begin{cases} -1 & \text{if } d_{m+1} = \beta - 1 \\ d_{m+1} & \text{if } d_{m+1} \in \{-1, 0, 1\} \\ 1 & \text{if } d_{m+1} = \beta + 1, \end{cases}$$

mapping the guard digit d_{m+1} into the set $\{-1, 0, 1\}$. For $\rho \in \{\beta, \beta + 1\}$, when $\|Q\| = \pm 2$, then, similarly,

$$d'_{m+1} = \|Q\| = \begin{cases} 2 & \text{if } d_{m+1} \in \{-\beta + 2, 2\} \\ -2 & \text{if } d_{m+1} \in \{-2, \beta - 2\}. \end{cases}$$

Note that the important cases of redundant binary representations fall in the class where $|\|Q\|| \leq 2$, here we shall see in the next section that two guard digits may be needed.

3 Redundant Binary Representations

There are two particularly interesting redundant representations for $\beta = 2$, the borrow-save, and the 2's complement carry-save representations, but we will first briefly consider the unsigned carry-save representation, only capable of representing nonnegative numbers.

Theorem 3. *Let $P = \sum_{i=\ell}^{m+k} d_i[2]^i \in \mathcal{P}[2, \{0,1,2\}]$ for $k \geq 1$ be a polynomial whose value $\|P\|$ is representable in $\mathcal{F}_{\ell,m}[2, \{0,1,2\}]$. With Q defined by $P = Q[2]^{m+1} + R$, then Q is either the zero polynomial or the polynomial $Q = 1 \cdot [2]^0$. In the last case, d_m must be zero since the value $\|P\|$ is representable, hence the polynomial can be rewritten by changing d_m into $d'_m = 2$ and discarding all higher order digits.*

Note that this result does not cover the 2's complement carry-save representation; we shall see later (Theorem 6) that two guard digits are sufficient and may be necessary, to determine the value of the prefix Q. Also, for the borrow-save (signed-digit) representations, two guard digits turn out to be sufficient.

3.1 Borrow-Save Representation

We will first show that two guard digits are necessary and sufficient to determine the value of Q in the case of signed-digit, base 2 (borrow-save) representation:

Theorem 4 (Guard digits for borrow-save). *Let $P = \sum_{i=\ell}^{m+k} d_i[2]^i \in \mathcal{P}[2, \{-1,0,1\}]$ for some $k \geq 2$ be a borrow-save polynomial whose value $\|P\|$ is representable in $\mathcal{F}_{\ell,m}[2, \{-1,0,1\}]$. With prefix Q defined by $P = Q[2]^{m+1} + R$, then the value of Q satisfies $\|Q\| \in \{-1,0,1\}$ and can be determined by:*

$$\|Q\| \equiv (2d_{m+2} + d_{m+1}) \pmod 4$$
$$\text{and } \|Q\| \in \{-1,0,1\}.$$

Thus, two guard digits are sufficient to determine $\|Q\|$, but two guard digits are also necessary unless the sign of the value is known.

Proof. By Lemma 1, for $\rho = 1$, $|\|Q\|| \leq 1$. Let S be defined by $Q = S[2]^2 + d_{m+2}[2] + d_{m+1}$, then again, by the lemma, $\|S\| \in \{-1,0,1\}$ and $\|Q\| = \|S\| \cdot 4 + d_{m+2} \cdot 2 + d_{m+1}$, hence, the result. To distinguish $d_{m+2}d_{m+1} = 01$ from $d_{m+2}d_{m+1} = \bar{1}1$, obviously two guard digits are needed, unless the sign of the value is known. □

Note that the conversion of the two guard digits could also be found by applying Theorem 2 for $\beta = 4$ by considering the digit pair $d_{m+2}d_{m+1}$ as an encoding of a base-4 digit. The value $\|Q\|$ of Theorem 4, equal to the converted guard digit d'_{m+1}, can also be expressed as

$$\begin{aligned}
\|Q\| = -1 &\Leftrightarrow (d_{m+2} \neq 0 \wedge d_{m+1} = 1) \\
&\quad \vee (d_{m+2} = 0 \wedge d_{m+1} = -1) \\
\|Q\| = 0 &\Leftrightarrow d_{m+1} = 0 \\
\|Q\| = 1 &\Leftrightarrow (d_{m+2} \neq 0 \wedge d_{m+1} = -1) \\
&\quad \vee (d_{m+2} = 0 \wedge d_{m+1} = 1),
\end{aligned}$$

or by this simple rule: Just negate the value of the digit d_{m+1} if d_{m+2} is nonzero and then discard all digits of weight higher than 2^{m+1}, leaving only one guard digit.

We would also like to eliminate that guard digit, i.e., convert it into $\mathcal{F}_{\ell,m}[2, \{-1,0,1\}]$, but, in general, this is not possible in constant time, as evident from Example 1 when $d_m = 0$, where logarithmic time is needed for such a rewriting. Hence, one may as well convert the polynomial into a nonredundant representation, in this case, probably into standard 2's complement.

Corollary 5. *Under the conditions of Theorem 4, for the case $k = 2$ of the addition of two borrow-save polynomials, the resulting polynomial $P = \sum_{i=\ell}^{m+2} d_i[2]^i$ can be converted into $P' = \sum_{i=\ell}^{m+1} d'_i[2]^i \in \mathcal{F}_{\ell,m+1}[2, \{-1,0,1\}]$ in constant time, where*

$$d'_i = d_i \quad \text{for } \ell \leq i \leq m$$
$$d'_{m+1} = \begin{cases} d_{m+1} & \text{if } d_{n+2} = 0 \\ d_{m+2} & \text{otherwise.} \end{cases}$$

This special case was stated without proof in [8] for correcting the output of a single signed-digit 4-to-2 adder, employing a selector on the output of the most significant adder. Alternatively, otherwise, in the case set, $d'_{m+1} = -d_{m+1}$, using the above rule.

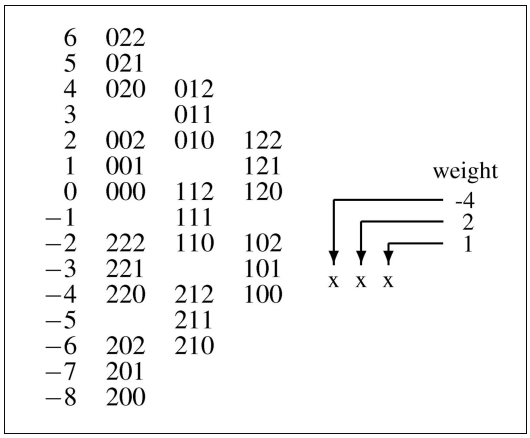
Note that these results imply that, in a multioperand addition, e.g., a tree structured adder, it is sufficient to perform the additions with two guard digits and only apply the correction in the final stage, leaving one guard digit, then possibly followed by a conversion into a nonredundant representation. There is no need to perform corrections in each addition.

Two guard digits can always be reduced to one if needed for reduced operand width, e.g., for use as a Booth recoded operand in a multiplication or, in general, when a redundant value may be used directly for some further calculations.

3.2 Two's Complement Carry-Save Representation

The combination of a redundant digit set with a radix-complement representation has turned out to be very useful for the case of radix 2, in the form of the 2's complement carry-save systems. Formally, the system is $\mathcal{F}^{2c}_{\ell,m}[2, \{-2,-1,0,1,2\}]$ with the restriction that negative digits are only allowed in position $m+1$ and only such that $d_{m+1} = -d_m$, i.e., $d_{m+1} \in \{-2,-1,0\}$. In practice, the digit d_{m+1} is not included in machine representations, hence, to illustrate some of the peculiarities, let us look at an example:

Example 2. The set $\mathcal{F}^{2c}_{0,2}[2, \{-2,-1,0,1,2\}]$ of three-digit, carry-save 2's complement integers and their (redundant) representations, without the presence of the leading nonpositive digit d_3 of value $d_3 = -d_2$, is:



Note that the sign of a number cannot be determined by the initial digit when it has the value 1 and also that zero is not uniquely represented.

But, the system is a subset of $\mathcal{F}_{\ell,m+1}[2,\{-2,-1,0,1,2\}]$, hence we can apply Lemma 1 since restricting the use of some of the digits cannot increase the bound on the prefix $\|Q\|$, which turns out to be $|||Q|||\leq 3$ since $\rho=3$. Observe, however, that this does not take the restrictions on the use of negative digits into account.

To be specific about the carry-save addition, we will describe it using three addition mappings

$$\alpha\colon\{0,1,2\}\times\{0,1,2\}\rightarrow\{0,1,2\}\times\{0,1\},$$
$$\gamma\colon\{0,1,2\}\times\{0,1\}\rightarrow\{0,1\}\times\{0,1\},$$

and $\xi\colon\{0,1\}\times\{0,1\}\rightarrow\{0\}\times\{0,1,2\}$, as described in Table 1, tables α, γ, and ξ, each specifying the output as a tuple cs, where c is the carry and s is the "place value."

Example 3. Adding 6 and 0 in some of the nonzero representations of zero in, respectivel, a 3, a 4, and a 5-digit system using the carry-save addition tables of Table 1 yields:

$\hat{\alpha}$	0	0	2	2		0	0	0	2	2		0	0	0	0	2	2
	$\bar{1}$	1	1	2		$\bar{1}$	1	1	1	2		$\bar{1}$	1	1	1	1	2
$\hat{\gamma}$	$\bar{1}$	1	1	0		$\bar{1}$	1	1	1	0		$\bar{1}$	1	1	1	1	0
	0	1	2			0	0	1	2			0	0	0	1	2	
$\hat{\xi}$	$\bar{1}$	0	1	0		$\bar{1}$	1	0	1	0		$\bar{1}$	1	1	0	1	0
	1	1	0			0	1	1	0			0	0	1	1	0	
	0	1	1	0		$\bar{1}$	2	1	1	0		$\bar{1}$	1	2	1	1	0

where the left-most digits are not part of the actual computation in the bounded length registers, but shown to illustrate the correct computation, and, hence, where the bounded length computations fail.

Note that the value 6 is representable in the 3-digit system, but only in the form 022, not in the form 110. If position $m+1$ is included, it is easy to see that all results correctly represent the sum 6; however, only in the 5-digit system is the result in $\mathcal{F}^{2c}_{\ell,m}[2,\{-2,-1,0,1,2\}]$ and only in this system is the string without the leading nonpositive digit a correct representation of the value 6. Note that it is not possible from the leading digit values to determine the sign of the number in this representation. The prefix string $\bar{1}12$ is just another representation of zero and the example illustrates that such leading guard digits must be included if the computation is performed without the digits in position $m+1$. Also note that the guard digits must be beyond *all possible representations* of the value, excluding prefixes of value zero, which is equivalent to saying that the guard digits must be beyond those needed to represent the value in *nonredundant 2's complement*.

TABLE 1
"Carry Save" Addition Tables

α	0	1	2
0	00	01	10
1	01	10	11
2	10	11	20

γ	0	1	2
0	00	01	
1	01	10	
2	10	11	

ξ	0	1	2
0	0	1	
1	1	2	
2			

The observations in the example tell us that it may be necessary with two guard digits, but this also turns out to be sufficient, as shown below.

Theorem 6 (Guard digits for 2's complement carry-save). *Let $P=\sum_{i=\ell}^{m+k+1}d_i[2]^i\in\mathcal{F}^{2c}_{\ell,m+k}[2,\{-2,-1,0,1,2\}]$ $k>2$, be a 2's complement carry-save polynomial whose value is representable nonredundantly in $\mathcal{F}^{2c}_{\ell,m}[2,\{-1,0,1\}]$. Let $Q\in\mathcal{F}^{2c}_{0,k-1}[2,\{-2,-1,0,1,2\}]$ be the prefix of P defined by $P=Q\,[2]^{m+1}+\sum_{i=\ell}^{m}d_i[2]^i$. Then, the value of Q can be determined by solving:*

$$\begin{aligned}&\|Q\|\equiv 2d_{m+2}+d_{m+1}\ (\mathrm{mod}\ 4)\\ &\textit{and}\ \|Q\|\in\{-2,-1,0\}\end{aligned}\tag{3}$$

and the polynomial

$$P'=\|Q\|\,[2]^{m+2}-\|Q\|\,[2]^{m+1}+\sum_{i=\ell}^{m}d_i[2]^i,$$

with a single guard digit $d'_{m+1}=-\|Q\|\in\{0,1,2\}$ has the same value as P.

Hence, two leading guard digits are sufficient to convert the representation of P into $P'\in\mathcal{F}^{2c}_{\ell,m+1}[2,\{-2,-1,0,1,2\}]$, where the conversion can be performed in constant time by a simple rewriting of the guard digits.

Proof. By Lemma 1, $|||Q|||\leq 3$, but, since $P\in\mathcal{F}^{2c}_{\ell,m}[2,\{-1,0,1\}]$,

$$\begin{aligned}|||Q|||2^{m+1}&\leq|||P|||+\left|\sum_{i=\ell}^{m}d_i2^i\right|\\&<2^m+2^{m+2},\end{aligned}$$

for $d_i \in \{0,1,2\}$, and, since $\|Q\|$ must be nonpositive, necessarily $\|Q\| \in \{-2,-1,0\}$. Let S be defined as the prefix of Q defined by $Q = S[2]^2 + d_{m+2}[2] + d_{m+1}$, then $\|S\| \in \{-1,0,1\}$ and $\|Q\| = 4\|S\| + 2d_{m+2} + d_{m+1}$, thus (3) holds. Hence, P can be represented in $\mathcal{F}^{2c}_{\ell,m+1}[2,\{-2,-1,0,1,2\}]$ by a simple rewriting of the digits d_{m+2} and d_{m+1} such that $d'_{m+2} = \|Q\| = -d'_{m+1}$. □

Note that it is not possible just to prepend the value of $\|Q\|$ as a digit in the set $\{-2,-1,0\}$ to the string $d_m d_{m-1}\cdots d_\ell$ to obtain a polynomial in $\mathcal{F}^{2c}_{\ell,m}[2,\{-2,-1,0,1,2\}]$ as this may violate the rule that $d_{m+1} = -d_m$. And, as we shall see below, a rewriting may require a change arbitrarily far to the right. Recall that it is required that the value $\|P\|$ is representable in nonredundant 2's complement, hence $-2^m \le \|P\| \le 2^m - 2^\ell$ or

$$-2^m \le \|P\| = \|Q\| \cdot 2^{m+1} + d_m 2^m + \sum_{i=\ell}^{m-1} d_i 2^i < 2^m. \quad (4)$$

If $\|Q\| = 0$, then $\|P\|$ must be nonnegative, hence changing the left bound to zero, then

$$0 \le d_m 2^m + \sum_{i=\ell}^{m-1} d_i 2^i < 2^m,$$

and, thus, d_m must be zero. For $\|Q\| = -2$, where $\|P\| < 0$, it follows similarly from (4) by changing the upper bound that

$$3 \cdot 2^m \le d_m 2^m + \sum_{i=\ell}^{m-1} d_i 2^i < 2^{m+2},$$

which implies that $2 \le d_m < 4$, hence $d_m = 2$. Thus, for $\|Q\| = 0, -2$, we have the correct value of d_m, but this is not always the case for $\|Q\| = -1$, where the sign of $\|P\|$ cannot be determined from that of $\|Q\|$. But, $-2^m \le \|P\| \le 2(2^{m-1} - 2^\ell)$, so

$$-2^m \le -2^{m+1} + (2d_m + d_{m-1})2^{m-1} + \sum_{i=\ell}^{m-2} d_i 2^i < 2^m,$$

where $0 \le \sum_{i=\ell}^{m-2} d_i 2^i < 2^m$, hence $\|Q\| = -1$ implies $1 \le 2d_m + d_{m-1} \le 5$. Now, there are three possible values of d_m, where $d_m = 1$ satisfies $d_m = -\|Q\|$, $d_m = 2$ is possible since this combination is equivalent to $d_{m+1} = d_m = 0$, and, finally, $d_m = 0$ is possible, together with $d_{m-1} = 1$ or 2. The combination $d_m d_{m-1} = 02$ can easily be changed into $d_m d_{m-1} = 10$ and $d_m d_{m-1} = 01$ is possible provided that $d_{m-1} = \cdots = d_{j+1} = 1$, with $d_j = 2$ for some $j \ge \ell$, because the string representation can then be changed into $\bar{1}100\cdots 0 d_{j-1}\cdots d_\ell$ representing the same value. However, this would require an arbitrary "look-ahead" into the trailing set of digits, which cannot be done in constant time.

To complete the picture, we will now show that, for $\|Q\| = -1$, if $d_m d_{m-1} = 01$ there is such an integer j, $\ell \le j \le m-1$, where $d_{m-1} = \cdots = d_{j+1} = 1$ with $d_j = 2$. From the bounds above, we have

$$2^m \le \sum_{j+1}^{m-1} 2^i + d_j 2^j + \sum_{i=\ell}^{j-1} d_i 2^i.$$

Now, assume there is no such j, i.e., $d_i \in \{0,1\}$ for $i = m-1, m-2, \cdots, \ell$. Then, there is a contradiction since the right-hand side of the inequality then has a value strictly smaller than 2^m. Hence, there must be such a $d_j = 2$ following the sequence of ones such that $d_j = 0$ would make the right-hand side even smaller.

We must then conclude that there is no constant time algorithm, mapping such a result with guard digits into the set $\mathcal{F}^{2c}_{\ell,m}[2,\{-2,-1,0,1,2\}]$. For practical purposes, the two guard digits should be retained during further computations until the result is to be mapped into a nonredundant representation.

Corollary 7. *Provided that the result of a computation performed in 2's complement, carry-save arithmetic is representable in ordinary nonredundant 2's complement representation in $\mathcal{F}^{2c}_{\ell,m}[2,\{-1,0,1\}]$, then it is sufficient to express the result with two guard digits as a carry-save number in $\mathcal{F}^{2c}_{\ell,m+2}[2,\{-2,-1,0,1,2\}]$, i.e., without the digits in position $m+3$ and higher.*

It is then possible in constant time to rewrite the guard digits such that the result is a correct polynomial P in 2's complement, carry-save representation with $P = \sum_{i=\ell}^{m+2} d_i[2]^i \in \mathcal{F}^{2c}_{\ell,m+1}[2,\{-2,-1,0,1,2\}]$, satisfying $d_{m+1} = -d_{m+2}$.

Conversion to remove both guard digits, i.e., mapping into the set $\mathcal{F}^{2c}_{\ell,m}[2,\{-2,-1,0,1,2\}]$, will, in general, require at least logarithmic time, i.e., time proportional to that required for conversion into nonredundant representation in $\mathcal{F}^{2c}_{\ell,m}[2,\{-1,0,1\}]$.

Conversion of the two guard digits into a single digit can, of course, be expressed in binary logic. We will here assume that a carry-save digit d_i is encoded as a bit pair (c_i, s_i) such that the value of the digit d_i is the arithmetic sum of c_i and s_i. The following two results were stated without proofs in [6].

Corollary 8. *The 2's complement polynomial $P = \sum_{i=\ell}^{m+k+1} d_i[2]^i \in \mathcal{F}^{2c}_{\ell,m+k}[2,\{-2,-1,0,1,2\}]$, $k > 2$, satisfying Theorem 6 can be converted into the 2's complement polynomial*

$$P' = -d'_{m+1}[2]^{m+2} + d'_{m+1}[2]^{m+1} + \sum_{i=\ell}^{m} d'_i[2]^i$$

in $\mathcal{F}^{2c}_{\ell,m+1}[2,\{-2,-1,0,1,2\}]$, where $d'_i = d_i$ for $i = \ell, \ell+1, \cdots, m$, and the rewritten digit d'_{m+1} is the arithmetic sum of c'_{m+1} and s'_{m+1}, as given by

$$\begin{aligned} s'_{m+1} &= s_{m+1} \oplus (c_{m+2} \oplus s_{m+2}) \\ c'_{m+1} &= c_{m+1} \oplus (c_{m+2} \oplus s_{m+2}), \end{aligned} \quad (5)$$

where $d_{m+1} \sim (c_{m+1}, s_{m+1})$ and $d_{m+2} \sim (c_{m+2}, s_{m+2})$.

Proof. Follows easily from the following table, expressing the relations between the values of $\|Q\|$, $\|S\|$ and the guard digits from the proof of Theorem 6,

$\|\|Q\|\|$	$\|\|S\|\|$	d_{m+2}	d_{m+1}	d'_{m+1}	s'_{m+1}	c'_{m+1}
0	0	0	0	0	0	0
	$\bar{1}$	2	0	0	0	0
	$\bar{1}$	1	2	0	0	0
-1	$\bar{1}$	1	1	1	0/1	1/0
-2	$\bar{1}$	1	0	2	1	1
	$\bar{1}$	0	2	2	1	1

using the appropriate encodings of the digits. □

Noll in [6] and Noll and De Man in [5] also provide a simplification in the case of a single 3-to-2 carry-save addition, as realized by an array of full-adders, showing a slightly modified (transistorized) full-adder, capable in a single most-significant guard digit position of performing the addition as well as the conversion. For completeness, here, we will restate and also prove his result.

Corollary 9. *Provided that the result of adding three nonredundant 2's complement numbers is representable in ordinary nonredundant 2's complement representation in $\mathcal{F}^{2c}_{\ell,m}[2,\{-1,0,1\}]$, the result can be calculated by an array of full adders, with the result represented as a carry-save, 2's complement polynomial*

$$P' = -d'_{m+1}[2]^{m+2} + d'_{m+1}[2]^{m+1} + \sum_{i=\ell}^{m} d_i[2]^i,$$

employing a single guard digit position (modified) full-adder producing the guard digit d'_{m+1}.

Let the result of an unmodified guard digit adder in position $m+1$ be the pair (c_{m+2}, s_{m+1}) and let c_{m+1} be the carry coming in from the adder in position m. Then, the encoding of the modified guard digit $d'_{m+1} \sim (c'_{m+1}, s'_{m+1})$ can be calculated by the modified adder in guard position $m+1$ as

$$\begin{aligned} s'_{m+1} &= s_{m+1} \oplus (c_{m+1} \oplus c_{m+2}) \\ c'_{m+1} &= c_{m+2}. \end{aligned} \tag{6}$$

Note. Noll [6] and Noll and De Man [5] present a full-adder and a slightly modified version calculating s'_{m+1} instead of s_{m+1}. It avoids the overhead of (6), by using the adder operands directly, together with c_{m+1}. Their full-adder is based on the expression $s = xyz + \bar{c}(x+y+z)$, where $c = xy + xz + yz$ is the carry-out, and the modified adder calculates $s'' = xyz + \bar{c}''(x+y+z)$ with c'' the carry-out from the previous position. This expression for s'' is equivalent to (6), except for some "dont-care" situations corresponding to real overflows. By using the modified adder in position $m+1$, then (6) can be calculated at practically the same cost in area and time.

Proof. Assume for the proof, initially, that the operands as well as the result are representable and that two guard digits are present in operands as well as in the output of a standard full-adder array and that the operands, say X, Y, and Z, are sign-extended into position $m+2$. Then, the addition can be described in digit string notation as:

$$\begin{array}{cccc|ccc} \cdots & x_{m+2} & x_{m+1} & x_m & \cdots & x_{\ell+1} & x_\ell \\ \cdots & y_{m+2} & y_{m+1} & y_m & \cdots & y_{\ell+1} & y_\ell \\ \cdots & z_{m+2} & z_{m+1} & x_m & \cdots & z_{\ell+1} & z_\ell \\ \hline \cdots & s_{m+2} & s_{m+1} & s_m & \cdots & s_{\ell+1} & s_\ell \\ c_{m+3} & c_{m+2} & c_{m+1} & c_m & \cdots & c_{\ell+1} & \end{array}$$

Due to the sign-extension of operands $x_{m+2} = x_{m+1}$, then $y_{m+2} = y_{m+1}$ and $z_{m+2} = z_{m+1}$, thus $s_{m+2} = s_{m+1}$. Inserting the latter identity in (5) and interchanging the role of c'_{m+1} and s'_{m+1}, then (6) follows and, thus, a single guard digit adder is sufficient. Implicitly, we have here assumed that the operands are representable in $\mathcal{F}^{2c}_{\ell,m}[2,\{-1,0,1\}]$. However, it is easily seen that if constants C_X, C_Y and C_Z, satisfying $C_X + C_Y + C_Z = 0$, are added to X, Y, and Z, respectively, the result would be the same. □

4 Conclusions

Given bounds on the result of a composite, additive fixed-point computation using a redundant number representation, it has been shown that it is sufficient to perform the operations with one or two leading guard digits, beyond those needed to represent the result. It is always possible in constant time to rewrite the leading digits of the result such that a single guard digit is sufficient. However, it has been shown for sign-digit and 2's complement carry-save representations that it may take at least logarithmic time to rewrite the result such that it contains no guard digits at all. Hence, removal of the leading guard digits should be postponed until the result anyway has to be converted to a nonredundant representation.

The results assure that, in binary multioperand addition or many other computations, it is sufficient to use one or two *leading guard digits* beyond the digits needed to represent the result, *even in the case that the result is known to have fewer digits than the operands.* And, in the case of the summation of k operands, when nothing in particular is known about these, it is sufficient with the guard digits beyond the at most $\log_\beta k$ extra digits needed to accommodate the worst-case growth of the sum compared to the operands.

Specific instances of the results presented in this paper must be well-known to anyone who has implemented systems employing redundant representations, like signed-digit and 2's complement carry-save, since such considerations seem unavoidable in many applications. Beyond the specific instances discussed in [6], [7], [8], we have not been able to find a general discussion about, and analysis of, the need for leading guard digits. It is our hope that the presented results may clarify the problems of "pseudo overflow" and aid future implementers.

REFERENCES

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

[2] A. Avizienis, "Signed-Digit Number Representations for Fast Parallel Arithmetic," *IRE Trans. Electronic Computers,* vol. 10, pp. 389-400, Sept. 1961.

[3] P. Kornerup, "Digit Selection for SRT Division and Square Root," *IEEE Trans. Computers,* vol. 54, no. 3, pp. 294-303, Mar. 2005.

[4] J.-M. Muller, *Elementary Function Evaluation: Algorithms and Implementation.* Boston, Basel, Berlin: Birkhäuser, 1997.

[5] T. Noll and E. De Man, "Anordnung zur Bitparallen Addition von Binärzahlen mit Carry-Save Überlaufkorrektur," European Patent EP 0 249 132 B1, Aug. 1993.

[6] T.G. Noll, "Carry-Save Architectures for High-Speed Digital Signal Processing," *J. VLSI Signal Processing,* vol. 3, pp. 121-140, 1991.

[7] B. Parhami, "On the Implementation of Arithmetic Support Functions for Generalized Signed Digit Number Systems," *IEEE Trans. Computers,* vol. 42, no. 3, pp. 379-384, Mar. 1993.

[8] D. Timmermann and B.J. Hosticka, "Overflow Effects in Redundant Binary Number Systems," *IEE Electronics Letters,* vol. 29, no. 5, pp. 440-441, Mar. 1993.

Peter Kornerup received the mag.scient. degree in mathematics from Aarhus University, Denmark, in 1967. After a period with the University Computing Center, beginning in 1969, involved in establishing the computer science curriculum at Aarhus University, he helped found the Computer Science Department in 1971. Through most of the 1970s and 1980s, he served as chairman of that department. Since 1988, he has been a professor of computer science at Odense University, now the University of Southern Denmark, where he also served a period as the chairman of the department. He spent a leave during 1975/1976 with the University of Southwestern Louisiana, Lafayette, four months in 1979 and shorter stays over many years with Southern Methodist University, Dallas, Texas, one month with the Université de Provence in Marseille in 1996 and two months with the Ecole Normale Supérieure de Lyon in 2001 and again in 2005. His interests include compiler construction, microprogramming, computer networks, and computer architecture, but, in particular, his research has been in computer arithmetic and number representations, with applications in cryptology and digital signal processing. Professor Kornerup has served on the program committees for numerous IEEE, ACM, and other meetings, in particular, he has been on the program committees for the Fourth through the 17th IEEE Symposia on Computer Arithmetic and served as program cochair for these symposia in 1983, 1991, and 1999, and is serving again for the 18th, to take place in 2007. He has been a guest editor for a number of journal special issues and served as an associate editor of the *IEEE Transactions on Computers* from 1991 to 1995. He is a member of the IEEE and a member of the IEEE Computer Society.

Jean-Michel Muller received the PhD degree in 1985 from the Institut National Polytechnique de Grenoble. He is Directeur de Recherches (senior researcher) at CNRS, France, and he manages the LIP laboratory (LIP is a joint laboratory of CNRS, the Ecole Normale Supérieure de Lyon, INRIA, and the Université Claude Bernard Lyon 1). His research interests are in computer arithmetic. Dr. Muller was program cochair of the 13th IEEE Symposium on Computer Arithmetic (June 1997), general chair of the 14th IEEE Symposium on Computer Arithmetic (April 1999), and is serving as the program cochair for the 18th, to take place in 2007. He served as associate editor of the *IEEE Transactions on Computers* from 1996 to 2000. He is a senior member of the IEEE and the IEEE Computer Society.