Proc. 36th Asilomar Conf. Signals, Systems, and Computers, Pacific Grove, CA, 2001 Nov. 4-7

Weighted Bit-Set Encodings for Redundant Digit Sets: Theory and Applications

Ghassem Jaberipur
Sharif Univ. of Technology
and Shahid Beheshti Univ.
Tehran 19847, Iran
jaberigh@mehr.sharif.ac.ir

Behrooz Parhami
Dept. Elec. & Computer Eng.
Univ. of California
Santa Barbara, CA 93106, USA
parhami@ece.ucsb.edu

Mohammad Ghodsi
Computer Engineering Dept.
Sharif Univ. of Technology
Tehran 11365, Iran
ghodsi@sharif.ac.ir

Abstract

This paper aims to fill the gap between theoretical studies of redundant number representation dealing with digit-level algorithms, without considering circuit-level details or impact of digit-set encodings, and implementation-oriented studies that typically focus on one particular digit-set encoding. We recognize that radices of practical interest are powers of two, giving each high-radix digit a weight that is a power of two. Furthermore, digit sets are typically encoded in such a way that each bit of the encoded form has a power-of-2 weight within the corresponding position. These observations lead us to define the class of weighted bit-set (WBS) encodings for redundant number systems and study the general properties of this class of representations. While by no means completely general, the class of WBS encodings includes virtually every implementation of redundant arithmetic that we have encountered, including those based on hybrid redundancy. We derive general conditions for a WBS encoding to be viable or efficient and describe how arithmetic operations can be performed on redundant numbers of this type using standard arithmetic components such as full/half-adders and multiplexers.

1. Introduction

Contributions to redundant number representation are of two main types. In abstract studies, arithmetic algorithms are presented in terms of digit-level operations, specifying how each result digit is derived from operand digits and auxiliary quantities such as interdigit transfers [Parh00]. Implementation-oriented studies, on the other hand, are often based on specific encodings for digit sets encountered in solving particular design problems; e.g., design of a high-speed 2's-complement full-tree multiplier [Taka85]. Some contributions of this latter type have dealt with limited classes of digit-set encodings without directly associating them with a specific design problem. Falling into the latter category are hybrid-redundant representation schemes [Phat94], [Phat01] and representation paradigms of high-radix signed digit numbers [Jabe02].

This paper aims to fill the gap between the aforementioned contributions. When in carry-free addition, the transfer digit t_{i+1}, going from digit position i to digit position $i + 1$, is specified in terms of $x_i + y_i$ (e.g., by supplying comparison constants and their associated selection intervals [Parh90]), no specific encoding of the digit set is implied; it is also not implied that one must actually add the digits x_i and y_i in the conventional sense and then compare the resulting sum to the boundary constants. Specifying t_{i+1} in terms of the relationship between $x_i + y_i$ and comparison constants is simply an intuitive way of defining $t_{i+1} = \tau(x_i, y_i)$, where τ is the transfer function. This is akin to defining a logic function via a logic expression; even though the expression directly corresponds to a logic circuit, one is free to choose any other implementation of the same logic function. Typically, choices for the comparison constants to determine t_{i+1} are flexible, thus leaving room for imprecise comparisons and a variety of circuit implementations based only on a subset of input bits.

We recognize that radices of practical interest are invariably powers of 2; thus, in practice, a redundant number can be viewed as a collection of digits, each weighted by a corresponding power of 2. Within each digit position, a digit value is also practically encoded as a collection of weighted bits. For example, the possibly asymmetric digit set $[-\alpha, \beta]$, with $\alpha \le 2^{\eta-1}$ and $\beta < 2^{\eta-1}$, might be encoded as an η-bit 2's-complement number, giving its bits the weights $-2^{\eta-1}, 2^{\eta-2}, \ldots, 2, 1$. As another example, BSD numbers [Aviz61] are commonly encoded by representing the position-i digit as two bits weighted -2^i and 2^i; this is known as the (n, p) encoding [Parh90]. Under these conditions (i.e., power-of-2 radix and weighted-bit-set representation of each digit), the number as a whole is encoded by a collection of bits, each weighted by a positive or negative power of two.

Definition 1 (WBS-encoded numbers): A *weighted bit-set* (WBS) encoding of width k is characterized by k integer pairs $(p_{k-1}, n_{k-1}), \ldots, (p_1, n_1), (p_0, n_0)$, where the representation has k radix-2 positions, indexed 0 to $k-1$, and digit position i $(0 \le i < k)$ of weight 2^i is comprised of n_i negatively weighted and p_i positively weighted bits. We require that $p_{k-1} + n_{k-1} > 0$. The most negative (positive) representable value of a WBS encoding is $-N$ (P), where $N = (n_{k-1} \ldots n_1 n_0)_{\text{two}}$ and $P = (p_{k-1} \ldots p_1 p_0)_{\text{two}}$. A given integer represented as $(v_{k-1} \ldots v_1 v_0)_{\text{two}}$, with $-n_i \le v_i \le p_i$, may have other WBS representations as well. ■

Definition 2 (Characterization of WBS encodings): The *encoding multiplicity* of position i in a WBS encoding is the total number $m_i = n_i + p_i$ of bits in that position. The ordered collection $m_{k-1} \ldots m_1 m_0$ of the positional multiplicities is the *multiplicity pattern* and $M = N + P$ is the *total multiplicity number*, which may be represented as the possibly redundant radix-2 number $(m_{k-1} \ldots m_1 m_0)_{\text{two}}$. Similarly, the ith *partial multiplicity number* M_i is $M_i = (m_{i-1} \ldots m_1 m_0)_{\text{two}} = N_i + P_i$ where $-N_i$ (P_i) is the most negative (positive) representable value by the rightmost i positions in the encoding. The total encoding cost is $E = \sum_{0 \le i < k} m_i$, leading to the encoding efficiency $e = \lceil \log_2(M+1) \rceil / E$. ■

Definition 3 (Negabits and posibits): We use *negabit* to denote a negatively weighted bit in [–1, 0] and *posibit* for a normal bit in [0, 1]. Graphically, ● (○) stands for posibit (negabit) in a natural extension of standard dot notation. ■

Example 1 (Familiar WBS-encoded numbers): The number representation systems whose descriptions follow are depicted in extended dot notation in Fig. 1. For unsigned carry-save representation, we have $m_i = p_i = 2$, $n_i = 0$, $\forall i$. Binary signed-digit (BSD) numbers with (n, p)-encoded digits have $n_i = p_i = 1$, $m_i = 2$. This represents, in effect, the 1's-complement encoding of the digit set [–1, 1]. Nonredundant 2's-complement number representation has $m_i = 1$, $\forall i$, $n_{k-1} = 1$, $p_i = 1$ for $i < k - 1$. For 2's-complement carry-save representation, we have $m_i = 2$, $\forall i$, with $n_{k-1} = 2$ and $p_i = 2$ for $i < k - 1$. In hybrid redundancy, with every fourth position being an (n, p)-encoded BSD digit, we have $m_i = p_i = 1$ and $n_i = 0$, except in positions whose index is 3 mod 4, for which $m_i = 2$, $n_i = 1$. ■

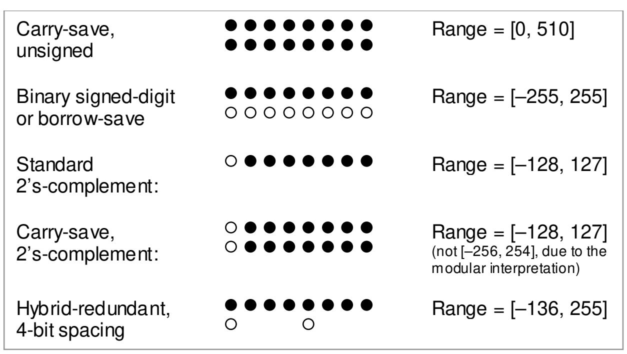

Fig. 1. Dot-notation representations for several familiar 8-position WBS-encoded number systems.

2. General WBS Encodings

In this section, we prove some general properties of WBS encodings. These general results are useful, because they cover, and tie together, numerous practical instances.

Definition 4 (Equivalent WBS encodings): WBS encodings representing precisely the same set of integer values are *equivalent*. *Strongly equivalent* WBS encodings are equivalent and have the same width k. ■

Example 2 (equivalent WBS encodings): The 8-position WBS encoding shown at the top of Fig. 2 is equivalent to the 7-position WBS encoding shown below it, and strongly equivalent to the 8-position encoding appearing at the bottom of Fig. 2. ■

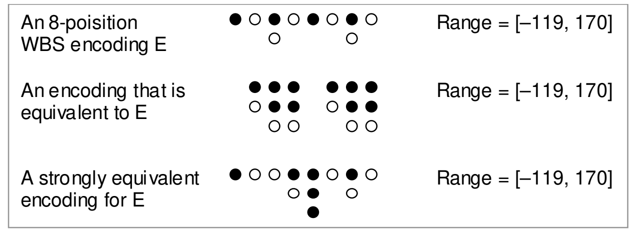

Fig. 2. Equivalent WBS encodings.

Theorem 1: An interval $[-N_k, P_k]$ of integer values containing $M_k + 1$ consecutive integers is representable by a WBS encoding with multiplicity pattern $m_{k-1} \ldots m_1 m_0$ iff for all i in the range $0 < i < k$, we have $M_i \ge 2^i - 1$.

Proof: The necessity part is easy to prove. If $M_i < 2^i - 1$ for some i, then positions 0 to $i - 1$ collectively represent fewer than 2^i distinct values. At least one of the 2^i mod-2^i equivalence classes must be unrepresented among these values. Given that bits in positions i and higher can only represent multiples of 2^i, there must be gaps in the representation. We prove the sufficiency part by induction on k. Recall that the multiplicity m is nonzero for the most-significant position of our postulated WBS representation. This leads to $m_0 > 0$, because either position 0 is the only position or else the condition of the theorem statement guarantees $M_1 = m_0 \ge 2^1 - 1$. The base case is $k = 1$; a one-position WBS representation with $m_0 > 0$ and clearly covers all integers from $-N_1 = -n_0$ to $P_1 = p_0$. Now suppose that the theorem holds for any WBS representation with $k - 1$ or fewer positions. Let a k-position WBS representation be obtained by extending a ($k - g$)-position representation, where $g \ge 1$, with $m_{k-1} > 0$ and $m_j = 0$ for $k - g \le j < k - 1$; i.e., the leftmost g components of multiplicity pattern are $m_{k-1} 0\, 0 \ldots 0$. Then, by our assumptions, $M_{k-1} = M_{k-2} = \ldots = M_{k-g} \ge 2^{k-1} - 1$. This implies that positions 0 to $k - 2$ can collectively represent a continuous interval of integers with at least 2^{k-1} consecutive values. These values combined with multiples of 2^k representable by the bit(s) in position $k - 1$ yield a continuous interval of integers overall. ■

Theorem 1 suggests that even though it is possible to avoid having any posibit or negabit in a particular position j of a WBS representation, doing so would require additional bits in lesser significant positions (two in position $j - 1$, four in position $j - 2$, etc.). Thus, for encoding efficiency, it is advantageous to enforce $m_i > 0$ for all i. On the other hand, replacement of a pair of bits of the same polarity in position j by one bit in position $j + 1$, through the substitutions outlined in Fig. 3, keeps $m_i \le 2$, and further improves encoding efficiency. These observations lead us to define the class of canonical WBS encodings.

Definition 5 (Canonical WBS encoding): A k-position WBS encoding is *canonical* iff $1 \le m_i \le 2$ for $0 \le i \le k - 2$. ■

Theorem 2: Any WBS encoding with the multiplicity pattern $m_{k-1} \ldots m_1 m_0$ satisfying $M_i \ge 2^i - 1$ for $0 < i < k$, and thus representing a continuous interval of integers in view of Theorem 1, is strongly equivalent to a unique k-position canonical WBS encoding.

Proof: We describe the process for deriving the canonical encoding from a given WBS encoding. Scan the multiplicities m_i from the right until you find $m_j \ge 3$ for some $j < k - 1$. If no such position exists, the encoding is already in the desired canonical form. If you find $m_j \ge 3$, take three of the bits in position j and make the substitution shown in Fig. 3. This does not change the set of values representable, and it reduces m_j by 2. Repeating this process eventually leads to $m_j \le 2$ for $0 \le j < k - 1$. To show that the resulting multiplicities satisfy $m_j \ge 1$, $0 \le j < k - 1$, we note that $M_j = (0m_{j-2} \ldots m_0)_{\text{two}}$ has a value of $2^j - 2$ when all the multiplicites assume the maximal value of 2. We can prove the uniqueness by contradiction. Suppose there are two equivalent but distinct canonical encodings and let l be the leftmost (most significant) position in which multiplicities differ. Then, the differece between the total multiplicity numbers of the two representations will be nonzero, because $M - M' \ge 2^l + (1\ 1 \ldots 1)_{\text{two}} - (2\ 2 \ldots 2)_{\text{two}} = 1$. ■

Corollary 1: A given WBS encoding is *redundant* iff in its equivalent canonical form, $m_j > 1$ for some $j < k$. ■

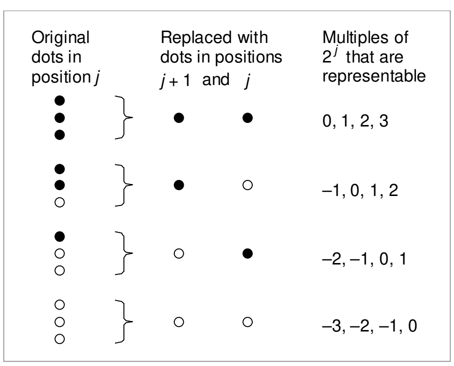

Fig. 3. Substitutions used in the proof of Theorem 2.

Theorem 3: Among all the strongly equivalent WBS encodings, the unique canonical WBS encoding has the highest encoding efficiency.

Proof: We show, by contradiction, that the encoding cost $E = \sum_{0 \le i < k} m_i$ is minimal for the unique canonical encoding. If the canonical encoding does not have the lowest cost among all strongly equivalent WBS encodings, uniqueness of the canonical encoding implies that the lowest-cost strongly equivalent encoding is noncanonical. This is impossible, however, because the process of transforming a WBS encoding to its canonical form (as described in the proof of Theorem 2) is solely composed of repeated applications of the substitutions shown in Fig. 3, and each substitution reduces the encoding cost E by 1. ■

3. Periodic WBS Encodings

Whereas arbitrary WBS encodings can be envisaged and used, circuit implementation in VLSI favors regularity in the number of bits associated with the various positions. Thus, we define the class of periodic WBS encodings.

Definition 6 (Periodic WBS encodings): A k-position WBS encoding is *periodic* iff there exist $h < k$ with $n_{i+jh} = n_i$ and $p_{i+jh} = p_i$ for all j; the smallest such h is the *period*. ■

Assuming k to be a multiple of h, a periodic WBS-encoded number represents a generalized signed-digit (GSD) number system in radix 2^h utilizing the digit set $[\alpha, \beta]$, with $\alpha = -(n_{h-1} \ldots n_1 n_0)_{\text{two}}$ and $\beta = (p_{h-1} \ldots p_1 p_0)_{\text{two}}$.

Given that full and half-adder cells, which are widely available and efficient, can be used to combine a set of bits with power-of-2 weights into another set of similarly weighted bits, periodic WBS encodings may be viewed as practically desirable GSD representations. In fact, all GSD representations that the authors have encountered in the literature are based on WBS encodings. Some examples are shown in Table I. For those digit sets in Table I that are symmetric, signed-magnitude encoding could conceivably be used, leading to a non-weighted-bit representation. However, we have been unable to find an actual implementation that is based on such a representation.

Table I. Some commonly used periodic WBS redundant number system encodings.

Digit set	Encoding name	# bits	Bit weights
$[-1, 1]$	(n, p)-encoded binary signed-digit	2	$1, -1$
$[-2, 1]$	2's-complement-encoded stored-double-borrow	2	$-2, 1$
$[-2, 2]$	Minimally redundant radix-4	3	$-2, 1, 1$
$[0, 2^h]$	Radix-2^h stored-carry	$h + 1$	$2^{h-1}, \ldots, 2, 1, 1$
$[-1, 2^h - 1]$	Radix-2^h stored-borrow	$h + 1$	$2^{h-1}, \ldots, 2, 1, -1$
$[-1, 2^h]$	Radix-2^h stored-carry-or-borrow	$h + 2$	$2^{h-1}, \ldots, 2, 1, 1, -1$
$[-2^{h-1}, 2^h - 1]$	Radix-2^h hybrid, with (n, p)-encoded BSD position	$h + 1$	$-2^{h-1}, 2^{h-1}, \ldots, 2, 1$
$[-2^h, 2^h - 1]$	Radix-2^h hybrid, with redundant digit values in $[-2, 1]$	$h + 1$	$-2^h, 2^{h-1}, \ldots, 2, 1$
$[-2^{h-1} - 1, 2^{h-1}]$	Radix-2^h stored-transfer, with transfers in $[-1, 1]$	$h + 2$	$-2^{h-1}, 2^{h-2}, \ldots, 2, 1, 1, -1$

Theorem 4: For an interval $[-N, P]$ of integers, that includes 0, and integer k satisfying $1 \le k \le \log_2(N + P + 1)$, there exists a unique k-position canonical WBS encoding representing exactly $[-N, P]$.

Proof: A trivial one-position WBS encoding with the given range has $n_0 = N$, $p_0 = P$, and $M = m_0 = N + P$. The unique k-position canonical encoding equivalent to the above can be easily derived by the construction of Theorem 2. ■

Corollary 2: For a radix-2^h GSD number system with digit set $[-\alpha, \beta]$, there is a unique periodic canonical WBS encoding with period h, where $1 \le h \le \log_2(\alpha + \beta + 1)$. ■

The next to last entry in Table I exemplifies a case where the bits in the encoding of adjacent digits overlap in terms of their weights. Such overlaps are avoidable by simply regrouping the bits. Figure 4 shows an example where the bits in a periodic WBS encoding with $h = 6$ are grouped in three different ways, each leading to a distinct digit set in radix-64 interpretation. Such variations are indeed useful for optimizing circuit implementations. Note that in the second and third groupings in Fig. 4, the boundary groups in the least- and most-significant end need special treatment, but this is generally not problematic. Note also that if two digits in $[-5, 65]$ are added, the obtained sum in $[-10, 130]$ is representable by the third digit set in Fig. 4. Hence, these two options in Fig. 4 collectively represent a *stored-transfer* scheme for carry-free addition [Jabe01]. This observation leads to the following general result.

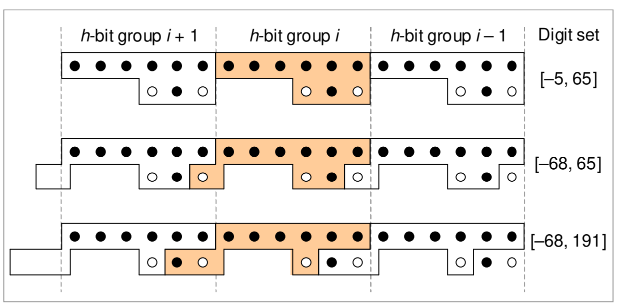

Fig. 4. Three different interpretations of the same periodic WBS encoding.

Theorem 5: Any stored-transfer scheme for radix-2^h GSD addition, where transfers are encoded as a set of posibits and negabits, can be explained in terms of bit grouping in a suitably chosen WBS encoding.

Proof: A stored-transfer scheme [Jabe01] is characterized by a main digit set $[a, b]$ and a transfer set $\{c_0, c_1, \ldots, c_{d-1}\}$, together constituting the radix-2^h digit set $[\alpha, \beta]$. If the transfer values are encoded as a set of posibits and negabits, as assumed, and the main digit set is encoded likewise, the overall representation is a periodic WBS encoding whose parameters m_j, n_j, and p_j within one period or radix-2^h digit, $0 \le j < h$, are obtained by adding the respective parameters of the main digit set and the transfer set. ■

4. Framework for WBS Arithmetic

Numbers with arbitrary digit sets can be added digitwise to produce a sum with a digit set whose range is the sum of the ranges of the operands. This wider digit set can be kept intact and the result used as an operand in further arithmetic operations. It is also possible to convert the wider digit set to another, more convenient, one for further processing. Often, however, it is required to obtain results with the same digit set as inputs. Such representationally closed arithmetic is desirable for reusability of the arithmetic cells and regularity in VLSI circuit implementation [Jabe00]. We note that when comparing a representationally closed scheme against a scheme that is not closed, fairness dictates that the overhead of conversion from the intermediate representation to the ultimate encoding be taken into account in any cost/speed comparison.

In circuit implementations, posibits are more easily dealt with than a mix of posibits and negabits, because they can be combined and regrouped using standard full adder, half-adder, and parallel counter cells. This motivates us to define 2's-complement-like WBS encodings in which negabits appear only in the most significant position $k - 1$, with all other positions holding only posibits.

Definition 7 (Two's-complement-like WBS encodings): A k-position WBS encoding is 2's-complement-like (2CL) if $m_i = p_i$, $0 \le i \le k - 2$. In a canonical 2CL-WBS encoding, we have $1 \le m_i = p_i \le 2$, $0 \le i \le k - 2$. ■

Theorem 6: For any k-position WBS encoding, there exists a unique $(k + 1)$-position canonical 2CL-WBS encoding. Furthermore, the latter can be constructed efficiently.

Proof: We describe the process for deriving the canonical 2CL-WBS encoding from a WBS encoding W. Consider a WBS encoding W' with the same multiplicity pattern as W, but with $p_i = m_i$, $\forall i$. Clearly, the range of W' is $[0, N + P]$. Now form $(k + 1)$-bit 2CL representation of the constant $-N$ with a single posibit in each of the positions 0 through $k - 1$ and one or more negabits in position k. Obtain the WBS encoding W'' by adding to each position of W' a posibit (one or more negabits in the case of position k) where the 2CL representation of $-N$ contains 1s. Clearly, the range of W'' includes $[-N, P]$. The desired canonical 2CL-WBS encoding is obtained by applying the first substitution of Fig. 3 to positions 0 to $k - 1$ that have more than 2 posibits until each of them holds 1 or 2 posibits. The process of converting a WBS number to a 2CL-WBS encoding can be implemented in parallel using time that is logarithmic in the depth d of the starting representation. ■

5. WBS Addition and Multiplication

In this section, we briefly describe algorithms for addition, subtraction, and multiplication of canonical 2CL-WBS numbers. Arithmetic algorithms for other operations, perhaps with a different encoding for each operand, can be developed by using either pre- or postoperation conversion. With preconversion, operands are changed to 2CL-WBS format before an operation. Postconversion allows an intermediate result (e.g., juxtaposition of bits for addition, or matrix of bitwise products for multiplication) to be formed based on the original operand bits.

Addition of two 2CL-WBS operands is performed by conceptually copying the bits of the 2-deep operands in the bit placeholders of a 4-deep WBS representation. This is then followed by conversion to canonical 2CL-WBS representation. Subtraction is similar, except that the posibits (negabits) of the second operand become negabits (posibits) in the intermdiate 4-deep result.

To multiply two 2CL-WBS encoded numbers, we might first derive a partial product bit matrix, and then reduce it through compression. The number of bitwise products to be dealt with can be 4 times greater than in standard binary multiplicaltion, given the depth of two for each operand. One way to reduce the complexity of our multiplier is to reduce the number of positions holding 2 posibits through partial carry assimilation. For example, if 4-bit segments of each 2-deep operand are combined to yield 5-bit binary numbers, with the MSB of one number aligned under the LSB of the next higher number, a radix-16 carry-save representation results for which efficient multiplication circuits have been studied [Ferg99].

6. WBS Coversions

Conversions of interest are: (1) 2's complement to WBS, (2) WBS to 2's complement, (3) One WBS form to another. Because the last category is quite varied, with conversion strategies differing depending on the source/target formats, we do not discuss it here in any detail except to note that any WBS-to-WBS conversion between formats of the same period can be viewed as digit-set conversion which can be performed in parallel and carry-free manner. A possible conversion strategy is to use the 2CL-WBS format as an intermediate format, thus needing to supply only a method for converting from 2CL-WBS to an arbitrary WBS.

Conversion from 2's-complement to 2CL-WBS is trivial, while conversion to a periodic WBS format can be done either directly as digit-set conversion or by first going to 2CL-WBS as an intermediate format. In either case, circuit implementation will be parallel and regular (consisting of identical cells), except in the most-significant end where the number sign must be processed differently. Conversion from WBS to 2's complement can similarly go through 2CL-WBS as an intermediate representation. The first phase (arbitrary WBS to 2CL-WBS) is carry-free whereas the second phase, like all redundant to nonredundant conversions, requires full carry propagation and is thus a logarithmic-time process at best.

7. Conclusion

In this paper, we introduced the class of weighted bit-set (WBS) redundant number representations that can lead to a fairly general strategy for obtaining efficient circuit implementations for redundant arithmetic using readily available, and highly optimized, building blocks developed for conventional binary arithmetic. For a given generalized signed-digit or hybrid-redundant representation, one can derive a suitable WBS encoding. The resulting encoding has the advantage that its intradigit propagation can be limited to posibit transfers, whereas in other instances, including hybrid redundancy, positive and negative carries coexist, leading to slower circuit implementations.

Extended WBS encodings that allow general two-valued digits, dubbed *twits* (e.g., having values in $\{-1, 1\}$, $\{0, 2\}$, or $\{0, -2\}$), are being investigated as a natural continuation of this work. This generalization not only enhances the encoding efficiency but also leads to speed gains in many instances. We have shown elsewhere that twits can be processed by essentially the same circuits that are applied to bits or negabits in this paper and are in the process of developing more complex twit-based arithmetic algorithms.

References

[Aviz61] Avizienis, A., "Signed-Digit Number Representations for Fast Parallel Arithmetic," *IRE Trans. Electronic Computers*, Vol. 10, pp. 389-400, Sep. 1961.

[Ferg99] Ferguson, M.I. and M.D. Ercegovac, "A Multiplier with Redundant Operands," *Proc. 33rd Asilomar Conf. Signals Systems and Computers*, Oct. 1999, pp. 1322-1326.

[Jabe00] Jaberipur, G., "High Radix Carry-Free Computer Arithmetic: Ph.D. Dissertation Proposal," Computer Engineering Dept., Sharif Univ. of Technology, Tehran, Iran, Mar. 2000.

[Jabe01] Jaberipur, G., B. Parhami, and M. Ghodsi, "A Class of Stored-Transfer Representations for Redundant Number Systems," *Proc. 35th Asilomar Conf. Signals Systems and Computers*, Nov. 2001, pp. 1304-1308.

[Jabe02] Jaberipur, G. and Ghodsi, M., "High Radix Signed Digit Number Systems: Representation Paradigms," to appear in *Scientia Iranica*.

[Parh90] Parhami, B., "Generalized Signed-Digit Number Systems: A Unifying Framework for Redundant Number Representations," *IEEE Trans. Computers*, Vol. 39, No. 1, pp. 89-98, Jan. 1990.

[Parh00] Parhami, B., *Computer Arithmetic: Algorithms and Hardware Designs*, Oxford, 2000.

[Phat94] Phatak, D.S. and I. Koren, "Hybrid Signed-Digit Number Systems: A Unified Framework for Redundant Number Representations with Bounded Carry Propagation Chains," *IEEE Trans. Computers*, Vol. 43, pp. 880-891, Aug. 1994.

[Phat01] Phatak, D.S. and I. Koren, "Constant-Time Addition and Simultaneous Format Conversion Based on Redundant Binary Representations," *IEEE Trans. Computers*, Vol. 50, No. 11, pp. 1267-1278, Nov. 2001.

[Taka85] Takagi, N., H. Yasuura, and S. Yajima, "High-Speed VLSI Multiplication Algorithm with a Redundant Binary Addition Tree," *IEEE Trans. Computers*, Vol. 34, No. 9, pp. 789-796, Sep. 1985.