High efficiency redundant binary number representations for parallel arithmetic on optical computers

G. A. DE BIASE, A. MASSINI

A family of redundant binary number representations, obtained by generalization of the RB (redundant binary) number representation, is introduced. All these number representations are suitable for optical computing and have properties similar to the RB representation. In particular, the p-RB (packed redundant binary) number representation introduced in this work has efficiency greater than both RB and MSD (modified signed digit) representations. With p-RB numbers the algebraic sum is always permitted in constant time for any efficiency value. p-RB representations also fit in a natural way the 2's complement binary number system. Symbolic substitution truth tables for the algebraic sum and several examples of computation are also given.

KEYWORDS: optical computing, redundant number representations, redundant binary numbers, parallel arithmetic units

Introduction

Efficient arithmetic units operating by symbolic substitution[1–4] can be obtained by using redundant number representations. In fact, by means of these representations it is possible to build totally parallel adders operating in constant time (the adding time is independent of the operand digit string length, N) by using symbolic substitution and, consequently, parallel multipliers operating in log-time[5,6]. In particular, the MSD (modified signed digit) number representation[7–12] has been widely studied for its suitability for digital optical computing.

In a recent paper[13], the redundant binary (RB) number representation was presented and studied in detail. The RB representation permits a two-step carry free algebraic sum and requires only two symbols $\{0, 1\}$ instead of three or more like the MSD representation[8]. With RB numbers the algebraic sum requires small truth tables using symbolic substitution or very simple hardware using other optical technologies[14,15]. In the present work a family of redundant binary number representations, obtained by generalization of the RB representation, is introduced.

All representations presented have properties similar to those or RB and are suitable for optical computing. In particular, the packed redundant binary (p-RB) representation, introduced here, has an efficiency greater than both the RB and MSD radix-two representation; with p-RB numbers, the algebraic sum is always permitted in constant time for any efficiency value. The p-RB representation fits, in a natural way, the 2's complement binary number system.

Redundant binary representations

Unsigned numbers

As presented in Ref. 13, an unsigned integer D given by

$$D = \sum_{i=0}^{N-1} a_i 2^{i-\lceil i/2 \rceil} \quad \text{with } N \text{ even} \tag{1}$$

where $a_i \in \{0, 1\}$ and the most significant digit is on the left end of the digit string, is in redundant binary number representation (the symbols $\lceil\ \rceil$ represent the rounding to the upper integer). For $i = \ldots 7, 6, 5, 4, 3, 2, 1, 0$, Equation (1) generates the following sequence of position weights

$$\begin{array}{cccccccccc} \ldots & 8 & 8 & 4 & 4 & 2 & 2 & 1 & 1 \\ & n & r & n & r & n & r & n & r \end{array}$$

The authors are in the Dipartimento di Scienze dell'Informazione, Università di Roma la Sapienza, Via Salaria 113, 00198 Roma, Italy.

0030–3992/94/04/0219–06 © 1994 Butterworth-Heinemann Ltd

Table 1. Example of unsigned RB1 numbers in canonical and redundant representation

D	bin.	RB1 representation							
		canonical	redundant						
0	000	000000							
1	001	000010	000001						
2	010	001000	000011	000100					
3	011	001010	000101	000110	001001				
4	100	100000	000111	001100	001011	010000			
5	101	100010	010010	001101	001110	010001	100001		
6	110	101000	010011	011000	001111	100100	100011	010100	
7	111	101010	010110	011001	100101	100110	011001	010110	010101
...	...	...	...	...	...	...	...	...	...

which characterizes this number representation. As can be seen, all position weights are doubled.

Another way to obtain a redundant binary number representation is[16]

$$D = \sum_{i=0}^{N-1} a_i 2^{i-\lfloor i/2 \rfloor} \quad \text{with } N \text{ even} \tag{2}$$

where again $a_i \in \{0, 1\}$ and the most significant digit is on the left end of the digit string (the symbols $\lfloor\ \rfloor$ represent the rounding to the lower integer). For $i = \ldots 7, 6, 5, 4, 3, 2, 1, 0$, the following sequence of position weights is generated

$$\begin{array}{cccccccc} \ldots & 16 & 8 & 8 & 4 & 4 & 2 & 2 & 1 \\ & n & r & n & r & n & r & n & r \end{array}$$

In this case, all position weights are doubled except the first and the last ones. The numbers generated by (1) and (2) will be called, respectively, in RB1 and RB2 representation. In each pair of digits of the same weight *nr*, the left and right digits are called, respectively, the *n* (normal) and *r* (redundant) digit. From (1) and (2) there follows that RB1 and RB2 representations of a number can be obtained from its binary representation by using the following simple recoding rules

$0 \rightarrow 00 \quad 1 \rightarrow 10$ for RB1

$0 \rightarrow 00 \quad 1 \rightarrow 01$ for RB2

namely: all *n* digits take the same value of the corresponding binary digit of the same weight, while all *r* digits are zeroed. The RB1 and RB2 numbers obtained in this way are in canonical form. Each RB number has several redundant representations, some examples of unsigned RB1 and RB2 numbers, in canonical and redundant representations, are shown in Tables 1 and 2. This coding operation is performable in parallel in one elemental logic step.

Signed numbers

In analogy with the 2's complement binary system, and in accordance with the encoding rules previously

Table 2. Example of unsigned RB2 numbers in canonical and redundant representation

D	bin.	RB2 representation			
		canonical	redundant		
0	000	000000			
1	001	000001			
2	010	000100	000010		
3	011	000101	000011		
4	100	010000	001000	000110	
5	101	010001	001001	000111	
6	110	010100	010010	001010	001100
7	111	010101	010011	001011	001101
...	...	...	...	...	...

presented, signed RB1 numbers are given by

$$D = -a_{N-1} 2^{N-1-\lceil (N-1)/2 \rceil} + \sum_{i=0}^{N-2} a_i 2^{i-\lceil i/2 \rceil} \quad \text{with } N \text{ even} \tag{3}$$

similarly, signed RB2 numbers are given by

$$D = -\sum_{i=N-2}^{N-1} a_i 2^{i-\lceil i/2 \rceil} + \sum_{i=0}^{N-3} a_i 2^{i-\lceil i/2 \rceil} \quad \text{with } N \text{ even} \tag{4}$$

Equations (3) and (4), derived from (1) and (2) respectively, guarantee that the canonical representation of a signed RB number can be obtained from the corresponding 2's complement binary number with the same recoding used for unsigned numbers. The recoding rules and the signed number definitions are slightly different from those given in Ref. 13. This is necessary for a correct generalization of the RB representation.

Algebraic sum

The redundant binary number representations permit a two-step totally parallel algebraic sum performable by symbolic substitution using rules defined by Truth Tables which act on *nr* digit pairs. Table 3 (T_{nr}) shows

Table 3. T_{nr}–symbolic substitution rule truth table for the algebraic sum of RB1 or RB2 numbers. This table acts on *nr* pairs; *u* and *l* indicate respectively the upper and lower row. The lower output pair is shifted left one position

	u			
l	00	01	10	11
00	00 00	10 00	00 01	10 01
01	00 01	10 01	00 10	10 10
10	00 01	10 01	00 10	10 10
11	00 10	10 10	00 11	10 11

these rules which, when applied in parallel on all pairs of two RB1 or RB2 numbers (operands), arranged on two superposed and aligned rows, give pairs of two output RB numbers, still on two superposed and aligned rows. The procedure is as follows.

Procedure 1
- The input is two superposed and aligned RB1 or RB2 numbers.
- Step 1 is the parallel application of the rules in Table T_{nr} on all *nr* pairs (this step generates an intermediate sum).
- Step 2 is the parallel application of the rules in Table T_{nr} on all *nr* pairs of the intermediate sum (this step generates the result).
- The output is the algebraic sum on the lower row and the zero is in canonical form on the upper row.

The correctness of Procedure 1 has been proved in Ref. 13. Figure 1 shows the alignment and truncation of RB1 and RB2 numbers for a correct application of table T_{nr}. In Fig. 2 examples of the algebraic sum of two RB1 and RB2 numbers are shown. As can be seen, the algebraic sum is completed in parallel in two elemental logic steps.

The algebraic sum on redundant binary numbers can also be performed by means of simpler tables acting on single digits. Table 4 presents these tables and the following procedure permits the algebraic sum.

Procedure 2
- The input is two superposed and aligned RB numbers.
- Step 1 is the parallel application of the rules in table T_{r1} on all *r* digits and the parallel application of the rules in table T_n on all *n* digits of the two input rows (this step generates an intermediate sum).
- Step 2 is the same as before (this step generates an intermediate sum).
- Step 3 is the same as before (this step generates an intermediate sum).
- Step 4 is the parallel application of the rules in table T_{r2} on all *r* digits and the parallel application of the rules in table T_n on all *n* digits of the two input rows (this step generates an intermediate sum).
- Step 5 is the same as before (this step generates an intermediate sum).
- Step 6 is the same as before (this step generates the result).
- The output is the algebraic sum on the upper row and the zero in canonical form on the lower row.

One can see that the algebraic sum is completed in six elemental logic steps. The correctness of this procedure can be proved following the same outline of the proof of correctness of Procedure 1 in Ref. 13: after Step 3, all upper *r* digits are always zeroed, the local carry is

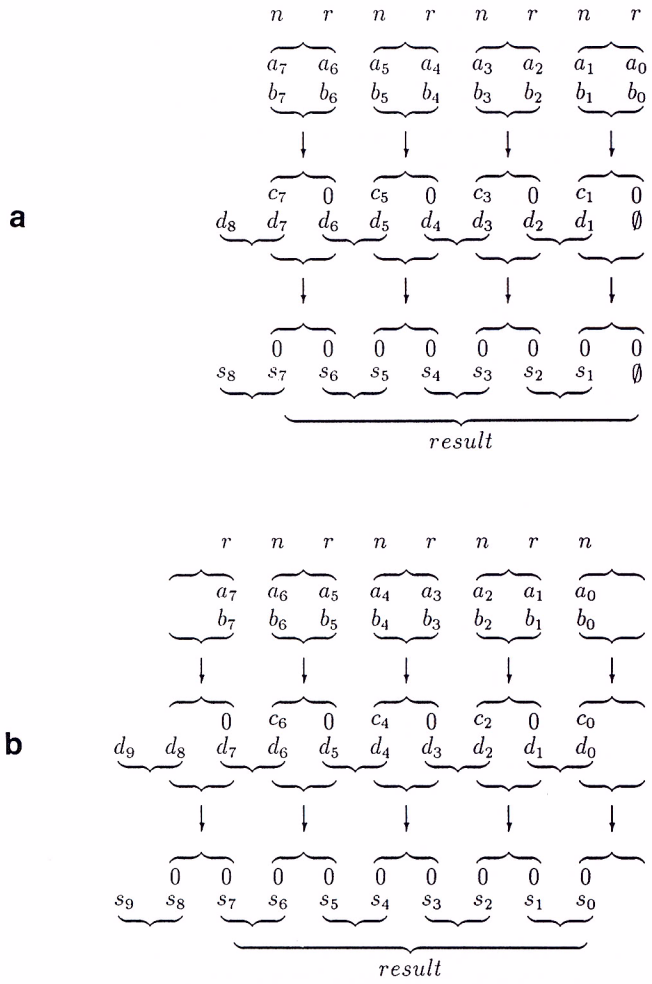

Fig. 1 Parallel application of the rules in Table 1 to perform algebraic sums by symbolic substitution: (a) on two RB1 numbers; (b) on two RB2 numbers. At each step the input and the output are on two strings: a_i and b_i are the bits of the input operands, c_i and d_i are the bits of the intermediate sum, and s_i are the bits of the result. The symbol $\emptyset$ is a padded zero

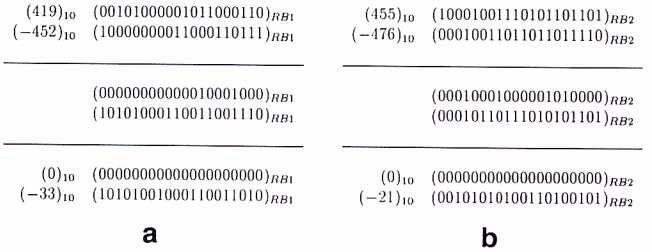

Fig. 2 Algebraic sum of two signed numbers applying Table T_{nr}: (a) on two RB1 numbers, (b) on two RB2 numbers

Table 4. Symbolic substitution rule truth tables for the algebraic sum of RB1 or RB2 numbers. These tables act on single *n* or *r* digits; *u* and *l* indicate respectively the upper and lower row. The lower output pair is shifted left one position

T_{r1}

l \ *u*	0	1
0	0 0	0 1
1	0 1	1 1

T_n

l \ *u*	0	1
0	0 0	1 0
1	1 0	1 1

T_{r2}

l \ *u*	0	1
0	0 0	1 0
1	1 0	1 1

positioned on these digits during the subsequent steps. Figure 3 shows examples of the algebraic sum of two RB1 or RB2 numbers using this procedure.

The additive inverse of an RB number is given in a similar way to that used in the 2's complement number system, taking into account the fact that the value of the negation of all the representations of the number $(0)_{10}$ is $(-2)_{10}$ in the RB1 representation and $(-3)_{10}$ in the RB2 representation. Then, the subtraction of two signed numbers is permitted. The additive inverse of RB numbers can be obtained in parallel in constant time (see Ref. 13).

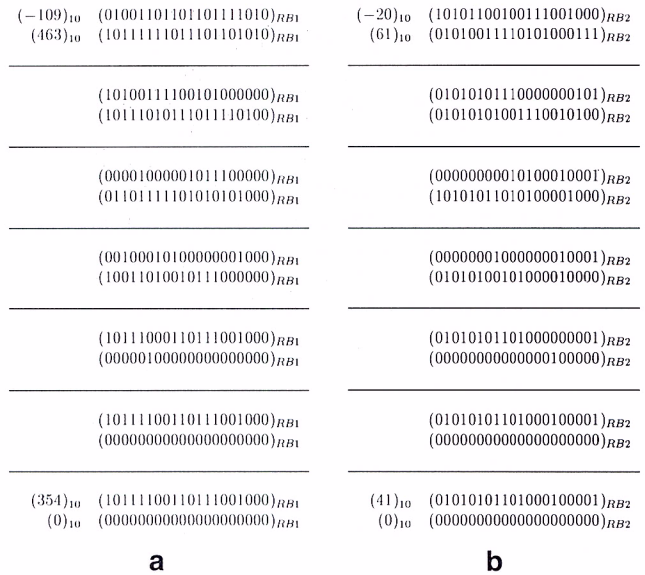

Fig. 3 Algebraic sum of two signed numbers applying tables acting on a single digit: (a) on two RB1 numbers; (b) on two RB2 numbers

Decoding

To compute the value of a RB1 or RB2 number, the use of (3) and (4) can generate values outside the range of the corresponding 2's complement binary number and the true sign of the number can be masked. To perform the correct decoding, signed numbers in redundant binary representations can be split into two binary numbers: the first is obtained by the n digits and the second by the r digits positioned on their correct weight. The binary sum of these two numbers gives the expected value in 2's complement binary representation. Only the $N/2$ least significant digits of the result must be considered (see Ref. 13).

Packed redundant binary number representation: *p*-RB

The efficiency, E, of a redundant number representation is the ratio between the number of bits needed in the binary number system and the number of bits needed in a redundant number representation to write the same value[13]. For both RB and MSD (radix-two) representations $E = 0.5$.

To increase the efficiency of redundant binary number representations the packing factor p can be introduced. By generalizing (1), an integer D can be given by

$$D = \sum_{i=0}^{N-1} a_i 2^{i-\lceil i/p \rceil} \qquad \text{where } p|N \tag{5}$$

where the digits are again $a_i \in \{0, 1\}$ and p is a positive integer. This number is in packed redundant binary (p-RB1) representation. In addition, by generalizing (2) one can obtain numbers in p-RB2 representation

$$D = \sum_{i=0}^{N-1} a_i 2^{i-\lfloor i/p \rfloor} \qquad \text{where } p|N \tag{6}$$

Equations (5) and (6) give the position weights of p-RB numbers. As an example, in the case of 4-RB1 numbers, for $i = \ldots 7, 6, 5, 4, 3, 2, 1, 0$, the weight sequence obtained from (5) is

$$\begin{array}{cccccccccc} \ldots & 32 & 16 & 8 & 8 & 4 & 2 & 1 & 1 \\ & n & n & n & r & n & n & n & r \end{array}$$

and the weight sequence obtained from (6) is

$$\begin{array}{cccccccccc} \ldots & 64 & 32 & 16 & 8 & 8 & 4 & 2 & 1 \\ & r & n & n & n & r & n & n & n \end{array}$$

For $p = 2$, the p-RB1 and p-RB2 representation coincide with the RB1 and RB2 representations respectively, consequently the p-RB numbers have properties very similar to those of the RB ones.

Algebraic sum

In analogy with (3) and (4), signed p-RB1 numbers are given by

$$D = -a_{N-1}2^{N-1-\lceil (N-1)/p \rceil} + \sum_{i=0}^{N-2} a_i 2^{i-\lceil i/p \rceil} \quad \text{where } p|N \tag{7}$$

and signed numbers in the p-RB2 representation are given by

$$D = -\sum_{i=N-2}^{N-1} a_i 2^{i-\lfloor i/p \rfloor} + \sum_{i=0}^{N-3} a_i 2^{i-\lfloor i/p \rfloor} \quad \text{where } p|N \tag{8}$$

The previously introduced tables T_n, T_{r1}, T_{r2} (acting on single digits) are a natural way for obtaining the algebraic sum of p-RB numbers. The procedure is as follows.

Procedure 3

- The input is two superposed and aligned p-RB numbers.
- Step 1 is the parallel application of the rules in table T_{r1} on all r digits and parallel application of the rules in table T_n on all n digits of the two input rows (this step generates an intermediate sum).
- Step 2 is the same as before (this step generates an intermediate sum).
- Step 3 is the same as before (this step generates an intermediate sum).
- Step 4 is the parallel application of the rules in table T_{r2} on all r digits and parallel application of the rules in table T_n on all n digits of the two input rows (this step generates an intermediate sum).
- Step 5 is the same as before (this step generates an intermediate sum).
- ...
- Step $p + 4$ is the same as before (this step generates the result).
- The output is the algebraic sum on the upper row and the zero in canonical form on the lower row.

Then, the algebraic sum is completed after $p + 4$ elemental logic steps. Figure 4 shows examples of the algebraic sum of two 4-RB1 and 4-RB2 numbers with this procedure.

Encoding and decoding of p-RB numbers

The encoding rules of p-RB numbers are a generalization of the rules of RB numbers. Any n digit takes the same value as the binary digit of the same weight, while all r digits are zeroed. These rules give, in parallel, in one elemental step, the canonical form of p-RB1 or p-RB2 numbers from its 2's complement binary representation.

To translate a p-RB number into its 2's complement binary corresponding number the same procedure of RB numbers is required. Namely: n digits and r digits (positioned on their correct weight) must be added, and only the least significant $N(p-1)/p$ digits of the result must be considered.

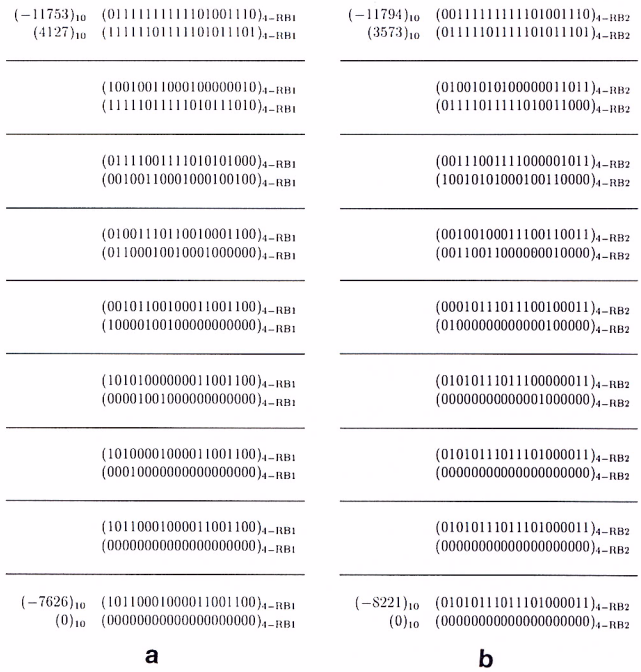

Fig. 4 Algebraic sum of: (a) two 4-RB1 numbers; (b) two 4-RB2 numbers

Additive inverse

The p-RB additive inverse can also be computed in analogy with the 2's complement binary number system. Let A be a bit string, N-bit wide, containing all 1s, and let p be the packing factor. String A is evidently the negation of a p-RB zero in canonical form. By decoding this string, using the rules presented in the previous sections, a 2's complement number C is obtained. This value must be used in the following way.

Procedure 4

- The input is a number in p-RB1 or p-RB2 representation.
- In Step 1 all digits of this number are negated.
- Step 2 is the algebraic sum between the RB1 and RB2 canonical form of $-C$ and the p-RB1 or p-RB2 number.
- The output is the additive inverse of the p-RB1 or the p-RB2 number.

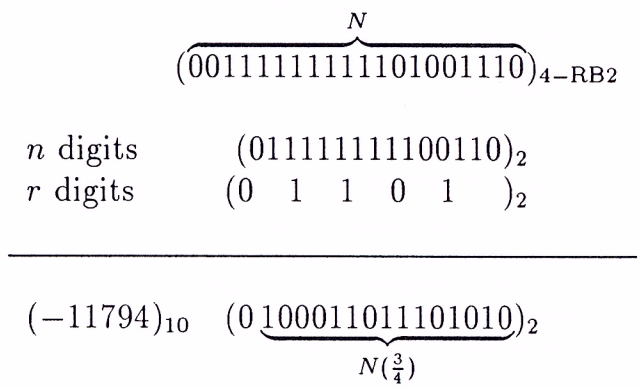

Fig. 5 Example of decoding of a signed 4-RB2 number. The least significant $N(\frac{3}{4})$ digits of the result give the 2's complement binary value