UUCS-89-013

The Set Theory of Arithmetic Decomposition

Tony M. Carter
Dept. of Computer Science
University of Utah
Salt Lake City, Utah

James E. Robertson
Dept. of Computer Science
University of Illinois
Urbana, Illinois

20 July 1989

VLSI Systems Research Group, Technical Report UUCS–89–013
University of Utah, Department of Computer Science

The Set Theory of Arithmetic Decomposition

TONY M. CARTER* (*carter@cs.utah.edu*)
University of Utah
Dept. of Computer Science
3190 Merrill Engineering Building
Salt Lake City, Utah 84112

JAMES E. ROBERTSON† (*rob@m.cs.uiuc.edu*)
University of Illinois
Dept. of Computer Science
1304 West Springfield
Urbana, Illinois 61801

Keywords: Computer Arithmetic, Set Arithmetic, Decomposition, Physical Design

Abstract. The Set Theory of Arithmetic Decomposition is a method for designing complex addition/subtraction circuits at any radix using strictly positional, sign-local number systems. The specification of an addition circuit is simply an equation that describes the inputs and the outputs as weighted digit sets. Design is done by applying a set of rewrite rules known as decomposition operators to the equation. The order in which and weight at which each operator is applied maps directly to a physical implementation, including both multiple-level logic and connectivity. The method is readily automated and has been used to design some higher radix arithmetic circuits. It is possible to compute the cost of a given adder before the detailed design is complete.

1 Introduction

The design of circuit structures for binary (radix 2) addition and subtraction is a mature discipline and is reasonably well understood for a wide variety of different types of adders. Among these are the ripple carry adders (two's complement, one's complement and sign-magnitude), the carry lookahead and block carry lookahead adders, the carry completion adder, the signed-digit adders, the carry save adder and the Wallace Tree adders. All of these are described in Hwang [16]. For the most part, practical designs have been done only for radix 2 arithmetic units since this represents the least complex design problem. The design of higher radix adders has not been considered in the design techniques commonly used today.

Of the techniques mentioned above, there are several which can be drawn together into a single design framework: two's complement ripple carry adders, carry

*This research was supported through DARPA contract number DAAK11-84-K-0017.

†This research was supported through NSF grant number MCS-83-08576.

save adders [17, 21], signed digit adders [1, 22] and Wallace Tree adders [3, 28]. Although these techniques for addition were developed for radix 2, they can easily be extended to arbitrarily high radix.

The Set Theory of Arithmetic Decomposition provides a uniform method of describing, specifying and designing adders (and subtracters) of these types at radix 2 or higher. It is based, as described below, on the notion of *set arithmetic*. With the theory, the design of circuits to implement the addition is reduced to applying a set of rewrite rules to an equation involving set addition and set scalar multiplication of digit sets that represent the inputs and outputs of the adder. Chow [10], Borovec [2], Ozarka [19] and Rohatsch [25] have all looked at the problem of designing redundant adders. The Set Theory of Arithmetic Decomposition provides a method for easily designing this class of adders without any extraordinary effort required to perform the logic design.

2 Definitions

The Set Theory of Arithmetic Decomposition deals with number representations that are *strictly positional* and *sign-local*. A strictly positional number representation is one in which the value of a number, whether positive or negative, is computed by a single formula. In sign-local representations, the sign digit does not affect the value of any other digit in the number. In sign-magnitude, the sign digit causes the value of every other digit to be negated. In diminished radix complement the sign digit affects at least the value of the least significant digit via the end around carry.

Table 1 shows the evaluation functions for four strictly positional number representations. We modify slightly the radix complement and diminished radix complement representations in that the most significant (or sign) digit can assume only the values 0 and $\overline{1}$ (we use $\overline{x} \equiv -x$) rather than 0 and $(r-1)$. In these equations, l is the lower bound and u is the upper bound and $u \geq l$. Thus, they may represent integers ($l = 0$), fractions ($u = 0$) or general fixed-point numbers ($l \neq 0$ and $u \neq 0$).

Each x_i in these equations is a digit that represents a set of integer values known as a *digit set*. A digit set is characterized by two parameters:

- δ, its *diminished cardinality* (one less than the number of its elements), and
- ω, its *offset* (the magnitude of its smallest element).

A digit set is denoted as $\langle \delta^{\omega} \rangle$. Thus, $\langle 1^0 \rangle$ represents the binary digit set $\{0,1\}$ and $\langle 9^0 \rangle$ represents the decimal digit set $\{0,1,2,3,4,5,6,7,8,9\}$. The notation $\langle \delta^{\star} \rangle$ denotes the set of digit sets whose elements are all the offset variations possible with that given δ. For example, $\langle 2^{\star} \rangle$ means $\{\{0,1,2\},\{\overline{1},0,1\},\{\overline{2},\overline{1},0\}\}$. Using the concepts of diminished cardinality and offset, a digit set D is defined as follows:

1. A digit set is a sequence of $\delta+1$ *consecutive* integers, $\{-\omega+0,\cdots,-\omega+\delta\}$.

Table 1: Strictly Positional Number Representations

Sign-Magnitude	$\mathbf{X} = (-1)^{x_u} \sum_{i=l}^{u-1} r^i x_i,$	$[x_u \in \{0, \overline{1}\}, x_i \in \{0, \cdots, (r-1)\}]$
Radix Complement	$\mathbf{X} = \sum_{i=l}^{u} r^i x_i,$	$[x_u \in \{0, \overline{1}\}, x_i \in \{0, \cdots, (r-1)\}]$
Dim. Radix Comp.	$\mathbf{X} = \left(\sum_{i=l}^{u} r^i x_i\right) - x_u,$	$[x_u \in \{0, \overline{1}\}, x_i \in \{0, \cdots, (r-1)\}]$
Signed-Digit	$\mathbf{X} = \sum_{i=l}^{u} r^i x_i,$	$[\lvert x_i \rvert \leq \alpha, \lceil \frac{r}{2} \rceil \leq \alpha \leq (r-1)]$

2. $\delta \geq 1$. At radix r, $\delta \leq (2r-2)$.
3. $\delta \geq \omega \geq 0$ which implies that $0 \in D$.
4. Where r is the radix, $\delta \geq r-1$ which implies that $(r \geq 2)$.

There are several other useful auxiliary definitions of specific digit sets:

1. If $\omega = 0$, the digit set is *normalized.*
2. If $(\delta \bmod 2 = 0)$ and $(\omega = \delta/2)$, the digit set is *symmetric.*
3. If $(\delta > r-1)$, the digit set is *redundant.*
4. The *negative* of a digit set is defined as $\overline{D} = \{-d \mid d \in D\}$.

Central to the Set Theory of Arithmetic Decomposition are two operations on sets of integers: *set addition* and *set scalar multiplication.* Given sets of integers D_i, set addition is defined as:

$$D_0 + D_1 = \{(d_0 + d_1) \mid d_0 \in D_0 \text{ and } d_1 \in D_1\}.$$

Based on integer addition, set addition is both associative and commutative. Given, in addition to D_i, a scalar s, set scalar multiplication is defined as:

$$sD = \{(s \cdot d) \mid s \text{ is an integer and } d \in D\}.$$

Based on integer multiplication, set scalar multiplication is associative, commutative and both right and left distributive over set addition. Set scalar multiplication takes precedence over set addition.

We also define the *decomposition relation*, $\Leftarrow$, which indicates that the right hand arithmetic set expression is to be transformed into the left hand expression. The $\Leftarrow$ relation is not reflexive; swapping the right and left hand sides of a set decomposition equation results in a different, inverse physical structure.

An *arithmetic set expression* is a collection of weighted digit sets involving set addition and set scalar multiplication. It is defined as

$$\sum_{i=0}^{N} s_i D_i$$

where s_i is a scalar and D_i is a digit set. An arithmetic set expression that represents a digit set is called a *composite* digit set and has

$$\delta_c = \sum_{i=0}^{N} s_i \delta_i \quad \text{and} \quad \omega_c = \sum_{i=0}^{N} s_i \omega_i .$$

Thus the set expression $8\langle 1^1\rangle + 4\langle 2^0\rangle + 2\langle 1^1\rangle + \langle 2^0\rangle$ has $\delta_c = 8{\cdot}1+4{\cdot}2+2{\cdot}1+2 = 20$, $\omega_c = 8\cdot 1 + 2\cdot 1 = 10$ and represents the $\langle 20^{10}\rangle$ digit set.

The notion of composite digit sets is of prime importance since it indicates that digit sets of high diminished cardinality can be represented by weighted sums of digit sets of lower diminished cardinality. This is clear when considering that a clean, strictly positional number of finite length represents a digit set. A four bit two's complement number represents the digit set

$$\langle 15^8 \rangle = \{\overline{8}, \cdots, 0, \cdots, 7\}$$

for which the representation as an arithmetic set expression is

$$8\langle 1^1\rangle + 4\langle 1^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle .$$

In fact, for binary addition and subtraction, all high-δ digit sets can be represented as weighted sums of binary ($\langle 1^\star\rangle$) and ternary ($\langle 2^\star\rangle$) digit sets. This is critical for the physical design method based upon the theory.

Another critical notion is the *information content* of digit sets and composite digit sets. This is defined to be the number of distinct signals (or bits) required in its physical realization. Table 2 contains the representations of the first few digit sets as composite digit sets composed only of a minimal number of binary and ternary digit sets and the corresponding information content.

In most cases, the *weighting radix* is 2; however it is possible to design structures with higher weighting radices. The selection of the weighting radix in an arithmetic unit represents a compromise between operational speed and the complexity and cost of design.

Table 2: Binary/Ternary Representations of Digit Sets of Higher-δ

Digit Set	Binary/Ternary Representation	Info. Content
$\langle 1^\star \rangle$	$\langle 1^\star \rangle$	1
$\langle 2^\star \rangle$	$\langle 2^\star \rangle$	2
$\langle 3^\star \rangle$	$2\langle 1^\star \rangle + \langle 1^\star \rangle$	2
$\langle 4^\star \rangle$	$2\langle 1^\star \rangle + \langle 2^\star \rangle$	3
$\langle 5^\star \rangle$	$2\langle 2^\star \rangle + \langle 1^\star \rangle$	3
$\langle 6^\star \rangle$	$2\langle 2^\star \rangle + \langle 2^\star \rangle$	4
$\langle 7^\star \rangle$	$4\langle 1^\star \rangle + 2\langle 1^\star \rangle + \langle 1^\star \rangle$	3
$\langle 8^\star \rangle$	$4\langle 1^\star \rangle + 2\langle 1^\star \rangle + \langle 2^\star \rangle$	4
$\langle 9^\star \rangle$	$4\langle 1^\star \rangle + 2\langle 2^\star \rangle + \langle 1^\star \rangle$	4
$\langle 10^\star \rangle$	$4\langle 1^\star \rangle + 2\langle 2^\star \rangle + \langle 2^\star \rangle$	5
$\langle 11^\star \rangle$	$4\langle 2^\star \rangle + 2\langle 1^\star \rangle + \langle 1^\star \rangle$	4
$\langle 12^\star \rangle$	$4\langle 2^\star \rangle + 2\langle 1^\star \rangle + \langle 2^\star \rangle$	5
$\langle 13^\star \rangle$	$4\langle 2^\star \rangle + 2\langle 2^\star \rangle + \langle 1^\star \rangle$	5
$\langle 14^\star \rangle$	$4\langle 2^\star \rangle + 2\langle 2^\star \rangle + \langle 2^\star \rangle$	6
$\langle 15^\star \rangle$	$8\langle 1^\star \rangle + 4\langle 1^\star \rangle + 2\langle 1^\star \rangle + \langle 1^\star \rangle$	4

3 Decomposition Equations

A *decomposition equation* has a digit set or composite digit set on both right and left hand sides of the decomposition relation. For example, we can specify a two digit radix-r complement adder as

$$r^2\langle 1^1 \rangle + r\langle (r-1)^0 \rangle + \langle (r-1)^0 \rangle \Leftarrow$$
$$(r\langle 1^1 \rangle + \langle (r-1)^0 \rangle) + (r\langle 1^1 \rangle + \langle (r-1)^0 \rangle) + \langle 1^0 \rangle$$

in which the final $\langle 1^0 \rangle$ digit set on the right hand side represents the carry in. Here, both right and left hand sides are composite digit sets. This can be rewritten so the left hand side is not a composite digit set as

$$\left\langle \left(r^2 + r(r-1) + (r-1) \right)^{r^2} \right\rangle \Leftarrow \langle (r + (r-1))^r \rangle + \langle (r + (r-1))^r \rangle + \left\langle 1^0 \right\rangle .$$

To facilitate negation, signed digit adders generally use symmetric digit sets so that negation can be done on a digit-by-digit basis. They are multi-staged structures specified by a set of arithmetic set equations. The general structure of an even radix

signed digit adder with minimal redundancy involves a set of equations of the form

$$\begin{aligned} r\langle 1^1\rangle + \langle r^0\rangle &\Leftarrow \left\langle r^{\frac{r}{2}}\right\rangle + \left\langle r^{\frac{r}{2}}\right\rangle \\ r\langle 1^0\rangle + \left\langle (r-1)^{\frac{r}{2}}\right\rangle &\Leftarrow \langle r^0\rangle + \langle 1^1\rangle + \left\{\left\langle (r-2)^{\left(\frac{r}{2}-1\right)}\right\rangle\right\} \\ \left\langle r^{\frac{r}{2}}\right\rangle &\Leftarrow \left\langle (r-1)^{\frac{r}{2}}\right\rangle + \langle 1^0\rangle \end{aligned}$$

whereas the general structure of an odd radix signed digit adder involves non minimal redundancy and a set of equations of the form

$$\begin{aligned} r\langle 1^1\rangle + \langle (r+1)^0\rangle &\Leftarrow \left\langle (r+1)^{\frac{(r+1)}{2}}\right\rangle + \left\langle (r+1)^{\frac{(r+1)}{2}}\right\rangle \\ r\langle 1^0\rangle + \left\langle r^{\frac{(r+1)}{2}}\right\rangle &\Leftarrow \langle (r+1)^0\rangle + \langle 1^1\rangle + \left\{\left\langle (r-2)^{\left(\frac{r-1}{2}\right)}\right\rangle\right\} \\ \left\langle (r+1)^{\frac{(r+1)}{2}}\right\rangle &\Leftarrow \left\langle r^{\frac{(r+1)}{2}}\right\rangle + \langle 1^0\rangle \end{aligned}$$

Additional redundancy in a signed digit number representation can reduce the number of stages, as in a radix-16 signed digit adder with $\langle 20^{10}\rangle$ digits.

$$\begin{aligned} 16\langle 2^1\rangle + \langle 16^8\rangle &\Leftarrow \langle 20^{10}\rangle + \langle 20^{10}\rangle \\ \langle 20^{10}\rangle &\Leftarrow \langle 16^8\rangle + \langle 2^1\rangle \end{aligned}$$

Here, the adder requires only two stages. Each of the digit sets with $\delta > 2$ can be represented by a composite digit set in which the weighting radix is 2.

It should be noted above that there are frequently cases where the diminished cardinality and/or offset of the right hand side is not equal to that of the left hand side. There are three cases to consider:

1. $\delta_{out} < \delta_{in}$.
2. $\delta_{out} = \delta_{in}$.
3. $\delta_{out} > \delta_{in}$.

In the first case, the structure is *unrealizable* since the input expression can represent more values than the output expression. The equations of the second case are called *decomposition equations* and are both *realizable* and *decomposable*. The equations of the third case are realizable but not decomposable since the output can represent more values than the input. Since decomposition deals with addition and subtraction, it is possible to transform equations of the third case ($\delta_{out} > \delta_{in}$) to the second case ($\delta_{out} = \delta_{in}$) by adding a *mythical input* to the right hand side of the equation such that $\delta_{out} = \delta_{in} + \delta_{myth}$ and $\omega_{out} = \omega_{in} + \omega_{myth}$. Mythical inputs always assume

a zero value and so do not change the result of an addition or subtraction. Since mythical inputs are not true inputs and always assume a value of zero, they may be used after decomposition to simplify the physical design.

The *inverse* of a decomposition equation is a decomposition equation in which the right and left hand sides have been switched. An equation that is decomposable ($\delta_{out} = \delta_{in}$) has an inverse that is also decomposable. An unrealizable equation ($\delta_{out} < \delta_{in}$) has an inverse that is realizable but not decomposable. A realizable but undecomposable equation ($\delta_{out} > \delta_{in}$) has a non-realizable inverse. Thus, decomposition equations that involve the use of a mythical input cannot be inverted.

A decomposition equation represents a structure for addition or inverse addition and a circuit that can be used to implement that structure. The right hand side of the decomposition equation represents the inputs to the structure while its left hand side represents the outputs of the structure. The logical design of the entire structure could be done by assigning logical variables to implement each of the inputs and outputs. The function is specified by the mapping of digit set values to boolean variables. Unfortunately, for many significant designs, the logical design problem is far too complex to be solved reasonably by known techniques. For example, in the design of a processor that uses radix-16 arithmetic with $\langle 20^{10} \rangle$ digit sets [4, 6, 7, 11, 23], the following decomposition equation is encountered.

$$\begin{gathered} 64\langle 2^1\rangle + 32\langle 1^1\rangle + 16\langle 2^0\rangle + 4\langle 2^1\rangle + 4\langle 1^1\rangle + 2\langle 1^0\rangle + \langle 2^0\rangle \Leftarrow \\ 32\langle 1^1\rangle + 32\langle 1^1\rangle + 16\langle 2^0\rangle + 16\langle 2^0\rangle + 8\langle 1^1\rangle + 8\langle 1^1\rangle + 8\langle 1^1\rangle + 8\langle 1^1\rangle + \\ 4\langle 2^0\rangle + 4\langle 2^0\rangle + 4\langle 2^0\rangle + 4\langle 2^0\rangle + 4\langle 2^1\rangle + 2\langle 1^1\rangle + 2\langle 1^1\rangle + \langle 2^0\rangle + \langle 2^0\rangle \end{gathered}$$

It represents a structure with 26 inputs and 11 outputs. Fully specifying 2^{26} input patterns, their mapping to 11 separate outputs and minimizing the resulting multiple-output function is a formidable problem not readily approachable using conventional techniques. Decomposition provides a method to design such circuits easily.

4 Decomposition (and Recomposition)

Decomposition is the process of applying *decomposition operators* to the right hand side of a decomposition equation to produce the left hand side. Recomposition is the inverse of decomposition in which *inverse decomposition operators* are applied to the right hand side of a decomposition equation to produce the left hand side. In decomposition, the information content of the right hand side is reduced to that of the left hand side. In recomposition, the information content of the right hand side is increased to that of the left hand side. Both decomposition and recomposition are based on two constraints:

1. Diminished cardinality is preserved. That is, $\delta_{out} = \delta_{in}$.
2. Offset is preserved. That is, $\omega_{out} = \omega_{in}$.

4.1 Decomposition Operators and Their Inverses

Decomposition operators and their inverses are specified by decomposition equations in which $\delta_{out} = \delta_{in}$. Each constitutes a rewrite rule that may be applied to the right hand side of a more complex decomposition equation and corresponds directly to a circuit module. The set of available decomposition operators depends on the set of allowed digit sets and the weighting radix.

To design non-redundant radix-r adders, we use the non-redundant $\langle (r-1)^* \rangle$ digit sets to represent digits of the operands and the result. We also need the $\langle 1^* \rangle$ digit sets to represent carries. In this case, only one class of decomposition operators is required, a *radix-r full adder*. In general, radix-r full adders are of the form

$$r\langle 1^* \rangle + \langle (r-1)^* \rangle \Leftarrow \langle (r-1)^* \rangle + \langle (r-1)^* \rangle + \langle 1^* \rangle .$$

For example, all that is required to implement all possible clean, strictly positional non-redundant adders in radix-2 is the set of radix-2 full adders:

$$2\langle 1^* \rangle + \langle 1^* \rangle \Leftarrow \langle 1^* \rangle + \langle 1^* \rangle + \langle 1^* \rangle$$

which is composed of the following four offset variations.

$$\begin{aligned}
2\langle 1^0 \rangle + \langle 1^0 \rangle &\Leftarrow \langle 1^0 \rangle + \langle 1^0 \rangle + \langle 1^0 \rangle \\
2\langle 1^0 \rangle + \langle 1^1 \rangle &\Leftarrow \langle 1^0 \rangle + \langle 1^0 \rangle + \langle 1^1 \rangle \\
2\langle 1^1 \rangle + \langle 1^0 \rangle &\Leftarrow \langle 1^0 \rangle + \langle 1^1 \rangle + \langle 1^1 \rangle \\
2\langle 1^1 \rangle + \langle 1^1 \rangle &\Leftarrow \langle 1^1 \rangle + \langle 1^1 \rangle + \langle 1^1 \rangle
\end{aligned}$$

These four radix-2 full adders are commonly used in array multipliers such as those proposed by Pezaris [20]. The number of radix-r full adders, N_{fa}, at radices $r > 2$ is

$$N_{fa} = 2 \cdot \binom{r+1}{2} = r^2 + r$$

so there are 12 distinct radix-3 full adders, 20 distinct radix-4 full adders and 272 distinct radix-16 full adders.

In redundant number representations such as signed digit, we must allow the use of all digit sets from $\langle 1^* \rangle$ up through $\langle (2r-2)^* \rangle$. The reason for this will shortly be evident. There are five forms of diadic decomposition operators. At weighting radix r, each form specifies a set of diadic operator classes each of which consists of a set of offset variants. The total number of diadic operator classes is given by

$$\binom{2r}{2} = 2r^2 - 3r + 1.$$

At $r = 2$ there are only 3 classes of diadic operators. At $r = 3$ there are 10 and at $r = 4$ there are 21. The five forms of diadic decomposition operators are as follows.

Partial Adders

$$\langle x^\star\rangle \Leftarrow \langle y^\star\rangle + \langle z^\star\rangle \quad \left[\begin{array}{l} 2r-2 \geq (x = y+z) \geq 2 \\ 2r-2 > y, z > 0 \end{array}\right]$$

Carry Generators

$$r\langle 1^\star\rangle + \langle (r-1)^\star\rangle \Leftarrow \langle y^\star\rangle + \langle z^\star\rangle \quad \left[\begin{array}{l} 2r-1 = y+z \\ 2r-2 \geq y, z > 0 \end{array}\right]$$

Super Carry Generators

$$r\langle 2^\star\rangle + \langle (r-1)^\star\rangle \Leftarrow \langle y^\star\rangle + \langle z^\star\rangle \quad \left[\begin{array}{l} 3r-1 = y+z; \\ 2r-2 \geq y, z > 0 \end{array}\right]$$

Redundant Carry Generators

$$r\langle 1^\star\rangle + \langle x^\star\rangle \Leftarrow \langle y^\star\rangle + \langle z^\star\rangle \quad \left[\begin{array}{l} 2r-2 \geq (x = y+z-r) \geq r \\ 2r-2 \geq y, z > 0 \end{array}\right]$$

Super Redundant Carry Generators

$$r\langle 2^\star\rangle + \langle x^\star\rangle \Leftarrow \langle y^\star\rangle + \langle z^\star\rangle \quad \left[\begin{array}{l} 2r-2 \geq (x = y+z-2r) \geq r \\ 2r-2 \geq y, z > 0 \end{array}\right]$$

Table 3 shows the diadic operator classes for radices 2, 3 and 4. The availability of $\langle 4^2\rangle$ for radix 3 implies that it is possible to build a signed digit adder at radix 3 for which negation can be done on a digit by digit basis.

A non-redundant, radix-r full-adder can be built by combining a partial adder and a carry generator of the following forms:

$$\begin{array}{lll} \mathrm{PA}_{r.(r-1)} & \langle r^\star\rangle & \Leftarrow \langle (r-1)^\star\rangle + \langle 1^\star\rangle \\ \mathrm{CG}_{(r-1).r} & r\langle 1^\star\rangle + \langle (r-1)^\star\rangle & \Leftarrow \langle r^\star\rangle + \langle (r-1)^\star\rangle \end{array}$$

Decomposition operators may themselves be hierarchically composed [18]. That is, a decomposition operator at a higher weighting radix can be implemented by specifying it using a lower weighting radix and decomposing. For example, the radix-4 super redundant carry generator

$$\mathrm{SRCG}_{12.6} = 4\langle 2^\star\rangle + \langle 4^\star\rangle \Leftarrow \langle 6^\star\rangle + \langle 6^\star\rangle$$

can be rewritten at a weighting radix of 2 as follows.

$$4\langle 2^\star\rangle + 2\langle 1^\star\rangle + \langle 2^\star\rangle \Leftarrow (2\langle 2^\star\rangle + \langle 2^\star\rangle) + (2\langle 2^\star\rangle + \langle 2^\star\rangle)$$

When decomposed, this generates a three level structure consisting of two $\mathrm{RCG}_{4.2}$s followed by a $\mathrm{CG}_{3.2}$ followed by a $\mathrm{PA}_{2.1}$. Thus decomposition can be carried out using higher radix decomposition operators and actually implemented using operators at a lower radix.

In general, for higher weighting radices, the decomposition operators for the partial adders $\langle x^\star\rangle \Leftarrow \langle y^\star\rangle + \langle z^\star\rangle$ can be replaced by a series of applications of

Table 3: Diadic Decomposition Operators at Radices 2, 3 and 4

$PA_{2.1}$	$\langle 2^* \rangle$	$\Leftarrow$	$\langle 1^* \rangle + \langle 1^* \rangle$
$CG_{3.2}$	$2\langle 1^* \rangle + \langle 1^* \rangle$	$\Leftarrow$	$\langle 2^* \rangle + \langle 1^* \rangle$
$RCG_{4.2}$	$2\langle 1^* \rangle + \langle 2^* \rangle$	$\Leftarrow$	$\langle 2^* \rangle + \langle 2^* \rangle$
$PA_{2.1}$	$\langle 2^* \rangle$	$\Leftarrow$	$\langle 1^* \rangle + \langle 1^* \rangle$
$PA_{3.2}$	$\langle 3^* \rangle$	$\Leftarrow$	$\langle 2^* \rangle + \langle 1^* \rangle$
$PA_{4.2}$	$\langle 4^* \rangle$	$\Leftarrow$	$\langle 2^* \rangle + \langle 2^* \rangle$
$PA_{4.3}$	$\langle 4^* \rangle$	$\Leftarrow$	$\langle 3^* \rangle + \langle 1^* \rangle$
$CG_{5.3}$	$3\langle 1^* \rangle + \langle 2^* \rangle$	$\Leftarrow$	$\langle 3^* \rangle + \langle 2^* \rangle$
$CG_{5.4}$	$3\langle 1^* \rangle + \langle 2^* \rangle$	$\Leftarrow$	$\langle 4^* \rangle + \langle 1^* \rangle$
$RCG_{6.3}$	$3\langle 1^* \rangle + \langle 3^* \rangle$	$\Leftarrow$	$\langle 3^* \rangle + \langle 3^* \rangle$
$RCG_{6.4}$	$3\langle 1^* \rangle + \langle 3^* \rangle$	$\Leftarrow$	$\langle 4^* \rangle + \langle 2^* \rangle$
$RCG_{7.4}$	$3\langle 1^* \rangle + \langle 4^* \rangle$	$\Leftarrow$	$\langle 4^* \rangle + \langle 3^* \rangle$
$SCG_{8.4}$	$3\langle 2^* \rangle + \langle 2^* \rangle$	$\Leftarrow$	$\langle 4^* \rangle + \langle 4^* \rangle$
$PA_{2.1}$	$\langle 2^* \rangle$	$\Leftarrow$	$\langle 1^* \rangle + \langle 1^* \rangle$
$PA_{3.2}$	$\langle 3^* \rangle$	$\Leftarrow$	$\langle 2^* \rangle + \langle 1^* \rangle$
$PA_{4.2}$	$\langle 4^* \rangle$	$\Leftarrow$	$\langle 2^* \rangle + \langle 2^* \rangle$
$PA_{4.3}$	$\langle 4^* \rangle$	$\Leftarrow$	$\langle 3^* \rangle + \langle 1^* \rangle$
$PA_{5.3}$	$\langle 5^* \rangle$	$\Leftarrow$	$\langle 3^* \rangle + \langle 2^* \rangle$
$PA_{5.4}$	$\langle 5^* \rangle$	$\Leftarrow$	$\langle 4^* \rangle + \langle 1^* \rangle$
$PA_{6.3}$	$\langle 6^* \rangle$	$\Leftarrow$	$\langle 3^* \rangle + \langle 3^* \rangle$
$PA_{6.4}$	$\langle 6^* \rangle$	$\Leftarrow$	$\langle 4^* \rangle + \langle 2^* \rangle$
$PA_{6.5}$	$\langle 6^* \rangle$	$\Leftarrow$	$\langle 5^* \rangle + \langle 1^* \rangle$
$CG_{7.4}$	$4\langle 1^* \rangle + \langle 3^* \rangle$	$\Leftarrow$	$\langle 4^* \rangle + \langle 3^* \rangle$
$CG_{7.5}$	$4\langle 1^* \rangle + \langle 3^* \rangle$	$\Leftarrow$	$\langle 5^* \rangle + \langle 2^* \rangle$
$CG_{7.6}$	$4\langle 1^* \rangle + \langle 3^* \rangle$	$\Leftarrow$	$\langle 6^* \rangle + \langle 1^* \rangle$
$RCG_{8.4}$	$4\langle 1^* \rangle + \langle 4^* \rangle$	$\Leftarrow$	$\langle 4^* \rangle + \langle 4^* \rangle$
$RCG_{8.5}$	$4\langle 1^* \rangle + \langle 4^* \rangle$	$\Leftarrow$	$\langle 5^* \rangle + \langle 3^* \rangle$
$RCG_{8.6}$	$4\langle 1^* \rangle + \langle 4^* \rangle$	$\Leftarrow$	$\langle 6^* \rangle + \langle 2^* \rangle$
$RCG_{9.5}$	$4\langle 1^* \rangle + \langle 5^* \rangle$	$\Leftarrow$	$\langle 5^* \rangle + \langle 4^* \rangle$
$RCG_{9.6}$	$4\langle 1^* \rangle + \langle 5^* \rangle$	$\Leftarrow$	$\langle 6^* \rangle + \langle 3^* \rangle$
$RCG_{10.5}$	$4\langle 1^* \rangle + \langle 6^* \rangle$	$\Leftarrow$	$\langle 5^* \rangle + \langle 5^* \rangle$
$RCG_{10.6}$	$4\langle 1^* \rangle + \langle 6^* \rangle$	$\Leftarrow$	$\langle 6^* \rangle + \langle 4^* \rangle$
$SCG_{11.6}$	$4\langle 2^* \rangle + \langle 3^* \rangle$	$\Leftarrow$	$\langle 6^* \rangle + \langle 5^* \rangle$
$SRCG_{12.6}$	$4\langle 2^* \rangle + \langle 4^* \rangle$	$\Leftarrow$	$\langle 6^* \rangle + \langle 6^* \rangle$

the decomposition operators for radix 2. This follows directly from Theorem 1, with $r = 2$. First, each of the digit sets $\langle x^* \rangle$, $\langle y^* \rangle$ and $\langle z^* \rangle$ can be represented by its unique binary representation with minimum information content. A series of applications of the binary decomposition operators can then be applied to transform the representation of $\langle y^* \rangle + \langle z^* \rangle$ into that of $\langle x^* \rangle$.

If the weighting radix r is an odd prime, the decomposition operators for the carry generators can be replaced by a series of applications of the binary decomposition operators, augmented by a radix r carry generator of the form

$$r\langle 1^* \rangle + \langle (r-1)^* \rangle \Leftarrow \langle (2r-2)^* \rangle + \langle 1^* \rangle .$$

Each use of the radix r carry generator may require the use of at most one inverse binary partial adder. Consider the expression

$$\langle (2r-1)^* \rangle + \langle s^* \rangle \Leftarrow \langle y^* \rangle + \langle z^* \rangle$$

with $2r - 1 + s = y + z$; $2r - 1 \leq y + z$. The binary representation of $\langle (2r-1)^* \rangle$ which has minimum information content has the least significant terms $2\langle 2^* \rangle + \langle 1^* \rangle$, since r is odd. If both y and z are even, the binary representation of $\langle y^* \rangle$ and $\langle z^* \rangle$ terminate in $\langle 2^* \rangle$. It is necessary to use an inverse partial adder of the form $\langle 1^* \rangle + \langle 1^* \rangle \Leftarrow \langle 2^* \rangle$ to represent the input $\langle (2r-1)^* \rangle$ for the radix r carry generator.

There also exist pathological cases for radices $r = 9 + 4i$ $(i = 0, 1, 2, \cdots)$, with $\langle y^* \rangle$ and $\langle z^* \rangle$ terminating in $4\langle 2^* \rangle + 2\langle 1^* \rangle + \langle 1^* \rangle$, which require use of the inverse binary carry generator $\langle 2^* \rangle + \langle 1^* \rangle \Leftarrow 2\langle 1^* \rangle + \langle 1^* \rangle$ followed by an inverse binary partial adder $\langle 1^* \rangle + \langle 1^* \rangle \Leftarrow \langle 2^* \rangle$. This particular combination of inverse binary operators is called a *fanout function*. For radices $r = 19 + 8i$ $(i = 0, 1, 2, \cdots)$, two such fanout functions are required. The general formula for these cases is

$$r = (2^{n+1} + 2^{n-1} - 1) + 2^n i \quad \begin{matrix} (n = 2, 3, 4, \cdots) \\ (i = 0, 1, 2, \cdots) \end{matrix}$$

for which 1 inverse partial adder and $n - 1$ fanout functions are required. Both $\langle y^* \rangle$ and $\langle z^* \rangle$ terminate in

$$2^n\langle 2^* \rangle + 2^{n-1}\langle 1^* \rangle + \cdots + 2\langle 1^* \rangle + \langle 1^* \rangle .$$

If the weighting radix r is composite, i.e., $r = p \cdot q$, it can be shown that radix p and radix q carry generator operators can be used for decomposition [18]. Consider the radix r carry generator

$$r\langle 1^* \rangle + \langle (r-1)^* \rangle \Leftarrow \langle (2r-2)^* \rangle + \langle 1^* \rangle .$$

With $r = p \cdot q$, the identities

$$\begin{aligned} r - 1 &= pq - 1 = q(p-1) + (q-1) \\ 2r - 1 &= 2pq - 1 = q[(2p-2) + 1] + (q-1) \end{aligned}$$

are substituted in the equation for the radix r carry generator to yield

$$q\,[p\,\langle 1^{\star}\rangle + \langle (p-1)^{\star}\rangle] + \langle (q-1)^{\star}\rangle \;\Leftarrow\; q\,[\langle (2p-2)^{\star}\rangle + \langle 1^{\star}\rangle] + \langle (q-1)^{\star}\rangle$$

in which the expressions in [ ] clearly represent a radix p carry generator.

As an example of the practical consequence of these observations, decomposition for radix 10 can be accomplished by augmenting the binary decomposition operators with the radix 5 carry generators [18]

$$5\,\langle 1^{\star}\rangle + 2\,\langle 1^{\star}\rangle + \langle 2^{\star}\rangle \;\Leftarrow\; 4\,\langle 1^{\star}\rangle + 2\,\langle 2^{\star}\rangle + \langle 1^{\star}\rangle\,.$$

4.2 Theorems About Decomposition

First, we must show that decomposition and recomposition can actually work. This is achieved through the following proofs. We stipulate beforehand that at radix r, digit sets have at most $\delta = (2r-2)$.

Theorem 1 (Fundamental Theorem of Decomposition) *Any representation ρ of a digit set of diminished cardinality δ_0 can be transformed through a series of decomposition operator applications to a unique representation ρ_u of that digit set which has minimum information content and a single digit set at each weight.*

Proof: The proof for this theorem is contained in the following two lemmas.

Lemma 1 (Uniqueness of Representation) *For every digit set and a given weighting radix r there exists a unique representation ρ_u that has a single digit set at each required power of the weighting radix.*

Proof: Given a digit set with diminished cardinality δ_0 and a weighting radix r, let the digit set be represented initially by a random distribution of δ_0 across a set of weights $r^i, (0 \le i \le m)$ where there are N_i $\langle 1^{\star}\rangle$ digit sets at weight r^i ($\sum_{i=0}^{i=m} N_i = \delta_0$). Note that the process could begin with a set of other (not $\langle 1^{\star}\rangle$) digit sets at each weight. We can easily get to the representation proposed above by applying a series of *inverse partial adders* at each weight to break each higher-δ digit set down to $\langle 1^{\star}\rangle$ digit sets.

For each weight r^i starting at $i = 0$, do the following until $N_i = 0$:

1. If $N_i > (2r-2)$
   (a) Combine $r-1$ of these N_i $\langle 1^{\star}\rangle$ digit sets using a succession of partial adders to form a single $\langle (r-1)^{\star}\rangle$ digit set at weight r^i.
   (b) Repeat the following $\lfloor N_i/(2r-2)\rfloor$ times:

i. Using partial adders, combine the $\langle (r-1)^\star \rangle$ digit set with $r-1$ more $\langle 1^\star \rangle$ digit sets to form a single $\langle (2r-2)^\star \rangle$ digit set.

ii. Combine the single $\langle (2r-2)^\star \rangle$ digit set with one $\langle 1^\star \rangle$ digit set using a $r\langle 1^\star \rangle + \langle (r-1)^\star \rangle \Leftarrow \langle (2r-2)^\star \rangle + \langle 1^\star \rangle$ carry generator.

iii. $N_{i+1} = N_{i+1} + 1$.

(c) There will now be at most $(r-1)$ $\langle 1^\star \rangle$ digit sets and exactly one $\langle (r-1)^\star \rangle$ digit set remaining at weight r^i. Partial adders combine these to form a single digit set at weight r^i which has $\delta \leq (2r-2)$.

2. Otherwise, $N_i \leq (2r-2)$. The N_i $\langle 1^\star \rangle$ digit sets are combined using partial adders to form a single digit set. Only at r^m can $N_m < (r-1)$.

At the conclusion of this algorithm, there is a single digit set with diminished cardinality of at least $(r-1)$ at each weight r^i except possibly the most significant. This shows that any representation ρ of a digit set can be reduced to a represention ρ_u with a single digit set at each weight.

Assume ρ_u is not unique, it would then be possible to leave out a digit set at a given weight, compensating by increasing the diminished cardinalities of any or all digit sets of other weights. ρ_u is of the form

$$r^m D_m + r^{m-1} D_{m-1} + \cdots + r^i D_i + r^{i-1} D_{i-1} + \cdots + D_0.$$

Let us assume that we can leave out the $r^i D_i$ term. If all less significant D_i have the maximum allowable diminished cardinality $(2r-2)$, then the maximum value representable by all less significant digit sets is

$$V_{lmax} = \sum_{j=0}^{i-1} \left(r^j (2r-2) \right) = (2r^i - 2)$$

and the smallest value representable by the more significant digit sets is

$$V_{umin} = r^{i+1}.$$

For the expression without $r^i D_i$ to represent a digit set, it must be the case that

$$V_{umin} \leq V_{lmax} + 1$$

But, for $(r^{i+1} = 2r^i - 1)$, r must be less than 2. This is not allowed (by the definition of a digit set), therefore the representation is unique.

Lemma 2 (Condition for Digit-Set-ness) *At weighting radix r and with digit sets of $\langle \delta_i^\star \rangle$, $(0 \leq i \leq m)$ limited such that $0 \leq \delta_i \leq (2r-2)$, an arithmetic set expression with a single digit set at each weight represents a digit set if and only if all $\delta_i \geq (r-1)$, $(0 \leq i < m)$.*

Proof: Consider an arithmetic set expression of the form

$$r^m \langle 1^\star \rangle + r^{m-1} \langle (r-1)^\star \rangle + \cdots + r^i \langle (r-1)^\star \rangle + r^{i-1} \langle (r-1)^\star \rangle + \cdots + \langle (r-1)^\star \rangle .$$

The largest value representable by the digit sets at weights r^0 through r^{i-1} is

$$((r^i - r^{i-1}) + (r^{i-1} - r^{i-2}) + \cdots + (r-1)) = r^i - 1$$

The smallest non-zero value representable by the digit sets at weights r^i through r^m is $r^i \cdot 1 = r^i$. So, the expression represents the digit set

$$\left\{0, \cdots, r^i - 1, r^i, \cdots, \left(r^{m+1} - 1\right)\right\} .$$

Now, consider the case where the digit set at weight r^{i-1} has diminished cardinality $r-2$ and all digit sets at weights lower than r^{i-1} have the maximum allowable diminished cardinality of $2r-2$. In this case, the set expression is

$$\begin{gathered} r^m \langle 1^\star \rangle + r^{m-1} \langle (r-1)^\star \rangle + \cdots + r^i \langle (r-1)^\star \rangle + \\ r^{i-1} \langle (r-2)^\star \rangle + r^{i-2} \langle (2r-2)^\star \rangle + \cdots + \langle (2r-2)^\star \rangle \end{gathered}$$

Here, the largest value representable by the digit sets at weights r^0 through r^{i-1} is

$$(r^i - 2r^{i-1}) + (2r^{i-1} - 2r^{i-2}) + \cdots + (2r-2) = r^i - 2$$

and the expression cannot represent the value $(r^i - 1)$ and does not represent a digit set. Therefore, for ρ_u to represent a digit set, every $\delta_i \geq (r-1)$.

Lemma 3 (Minimum Information Content) *The unique representation, ρ_u, has minimum information content.*

Proof: Information content is defined as the number of boolean variables required to implement a digit set representation. We have already shown that the diminished cardinality of a digit set in ρ_u must be at least $r-1$ and have stipulated that it can be no larger than $(2r-2)$.

Assume that ρ_u does not have minimum information content. It must then be possible to represent a single digit set as a sum of two digit sets for which the implementation would require fewer bits. The number of bits required to represent a digit set is $\lceil \log_2 \delta \rceil$. So, if ρ_u does not have minimum information content then

$$\lceil \log_2 x \rceil > \lceil \log_2 y \rceil + \lceil \log_2 z \rceil$$

where $x = y + z$. First, assume that $2^j \leq x \leq 2^{j+1}$. The information content, $j+1 \leq I_x \leq j+2$. Let us then represent x using $y = 1$ and $z = x - 1$. So,

$I_y = 1$ and $j \leq I_z \leq j+1$. The information content of the output in this case is $j+1 \leq I_y + I_z \leq j+2$ which is the same range as for x. In fact, if $x > 2^j$ then $I_z = j+1$ and $I_y + I_z = j+2$ whereas $I_x = j+2$ only when $x = 2^{j+1}$, so when $y = 1$ it is not possible to represent x with fewer bits. We cannot lower the information content below one bit, so if we wish to reduce the total information content of $y + z$ we must reduce the information content of z. This may be done by setting $z = \lfloor \frac{x}{2} \rfloor - 1$ for which $j - 1 \leq I_z \leq j$. But, when we do this, we must set $y = \lceil \frac{x}{2} \rceil + 1$ for which $j \leq I_y \leq j+1$ so that $2j - 1 \leq I_y + I_z \leq 2j + 1$. If x does not have minimum information content then $(j+1) > (2j-1) \Rightarrow (j = 1)$ or $(j+2) > (2j+1) \Rightarrow (j = 0)$. In the first case I_x is at the absolute minimum of 1 bit so no representation with less information content exists. The second case cannot occur since it must be that $I_x > 0$. Therefore, by contradiction, the representation ρ_u must have minimum information content.

Theorem 2 (Fundamental Theorem of Recomposition) *Any representation ρ_1 of a digit set can be converted to any other representation ρ_2 of that digit set.*

Proof: By theorem 1, both representations ρ_1 and ρ_2 are decomposable to the same unique minimal representation ρ_u. Since there exist (by definition) inverses to each of the operators applied during decomposion, ρ_1 can be decomposed to ρ_u and then the steps used to decompose ρ_2 into ρ_u can be followed in inverse order from ρ_u using these inverse operators to arrive at ρ_2.

4.3 Verification of Digit Set-ness

There are two methods of determining if an arithmetic set expression represents a digit set. The first and most efficient is suggested in lemma 2. If the terms of an arithmetic set expression are arranged in order of ascending weight

$$w_m \langle \delta_m^\star \rangle + \cdots + w_i \langle \delta_i^\star \rangle + w_{i-1} \langle \delta_{i-1}^\star \rangle + \cdots + \langle \delta_0^\star \rangle$$

then it represents a digit set if

$$\forall i > 0,\ \delta_{i-1} \geq \frac{w_i}{w_{i-1}} - 1.$$

The second is to compute the values of δ and ω for the expression as if it did indeed represent a digit set, checking that $\delta > 0$ and $0 \leq \omega \leq \delta$. Then perform the complete set addition and verify that the resulting set contains all integers between $-\omega$ and $-\omega + \delta$.

4.4 Determination of Mythical Inputs

When a decomposition equation has $\delta_{out} > \delta_{in}$, it can be made decomposable by adding a mythical input (not necessarily a digit set) whose value is always zero

and therefore does not affect the result of addition or subtraction. Mythical inputs are selected to have minimum information content so that they impact the final implementation as little as possible. Our algorithm (which follows) involves using as much of the diminished cardinality and offset at as high a weight as possible. For example, it is better to use an $8\langle 1^0\rangle$ digit set with one bit of information content than a $4\langle 2^0\rangle$ digit set which has two bits of information content.

1. Let $\delta_{k+1} = (\delta_{out} - \delta_{in})$.
   Let $\omega_{k+1} = (\omega_{out} - \omega_{in})$.
2. If $(0 \geq \delta_{k+1}$ or $0 > \omega_{k+1}$ or $\delta_{k+1} < \omega_{k+1})$ then the determination of mythical inputs is not possible.
3. Let $k = \lfloor \log_r(\delta_{k+1}) \rfloor$.
4. Until $(k = -1$ or $\delta_{k+1} = 0)$ do the following:
   (a) $\omega_m = \left\lfloor \frac{\omega_{k+1}}{r^k} \right\rfloor$.
   (b) $\delta_m = \left\lfloor \frac{\delta_{k+1} + r^k \omega_m - \omega_{k+1}}{r^k} \right\rfloor$.
   (c) if $(\delta_m > 0)$ add $r^j \langle \delta_m^{\omega_m} \rangle$ to the mythical input
   (d) $\delta_k = \delta_{k+1} - \delta_m$
   (e) $\omega_k = \omega_{k+1} - \omega_m$
   (f) $k = k - 1$

This algorithm generates mythical inputs with minimum information content, and with offset used up at as high a weight as possible. For the purposes of logical design, mythical inputs may be used to simplify some operators, often reducing them to a set of wires (possibly involving fanout) with no logic required.

4.5 Information Loss Tables

Decomposition is achieved by applying decomposition operators at a given weight and level. The best design will use the fewest operators and levels possible. The design process can be captured in a form known as an *information loss table*, the generation of which is readily automated.

The form for an information loss table is shown in table 4 which is developed for $r = 2$. There are columns for each power of the weighting radix. Under each of these columns there are up to $N = \sum_{(i=1)}^{(2r-2)}(i + 1)$ subcolumns, one for each allowable digit set, including offset variations. The first line tallies the digit sets in the input representation of the digit set, the second holds the digit sets in the

Table 4: An Information Loss Table Skeleton

Level	**Operator** (#,2_i) **Name**	λ	**2** $\langle 2^2\rangle\langle 2^1\rangle\langle 2^0\rangle\langle 1^1\rangle\langle 1^0\rangle$	**1** $\langle 2^2\rangle\langle 2^1\rangle\langle 2^0\rangle\langle 1^1\rangle\langle 1^0\rangle$
Input Myth. 0				
1 • • • N				

mythical input (if any) and the third the sum of the first two. All subsequent rows in the table contain intermediate representations of the digit set that result from application of the decomposition operator(s) noted by name (as contained in table 3) in the "operator" column. The column labelled λ holds the value of the current information content.

The following sections contain some examples of decomposition. The information loss table for each design contains columns at each weight for only those digit sets actually used in the design. The information loss table is also rendered as a block diagram showing the interconnection of circuit modules required to implement the arithmetic function described. We have limited the use of offset in the examples to reduce the width of information loss tables. The operator names (as contained in table 3) are followed by a list of the digit set offsets. For example,

$$\mathrm{RCG}_{4.2}(1011) \equiv 2\langle 1^1\rangle + \langle 2^0\rangle \Leftarrow \langle 2^1\rangle + \langle 2^1\rangle$$

4.6 A Radix-2 Parallel Counter with Eleven Inputs

The following decomposition equation for a radix-2 parallel counter ([15, 27]) with eleven inputs results in the information loss table and equivalent block diagram shown in figure 1.

$$4\langle 2^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle \Leftarrow$$
$$\langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle .$$

The design has seven levels and uses fifteen decomposition operators. The initial input is $\langle 11^0\rangle$ and the output is $\langle 11^0\rangle$ so no mythical inputs are required. Decomposition begins with eleven $\langle 1^0\rangle$ at weight $2^0 = 1$. The succession of decomposition

Lev	Operator (#,W) Name	λ	4 $\langle 2^0\rangle\langle 1^0\rangle$	2 $\langle 2^0\rangle\langle 1^0\rangle$	1 $\langle 2^0\rangle\langle 1^0\rangle$
I		11			11
M					
0		11			11
1	(5,1) $PA_{2.1}(000)$	11			5 1
2	(2,1) $RCG_{4.2}(0000)$				
	(1,1) $CG_{3.2}(0000)$	8		3	2 1
3	(1,2) $PA_{2.1}(000)$				
	(1,1) $RCG_{4.2}(0000)$	7		1 2	1 1
4	(1,2) $PA_{2.1}(000)$				
	(1,1) $CG_{3.2}(0000)$	6		2 1	1
5	(1,2) $RCG_{4.2}(0000)$	5	1	1 1	1
6	(1,2) $CG_{3.2}(0000)$	4	1	1	1
7	(1,4) $PA_{2.1}(000)$	4	1	1	1

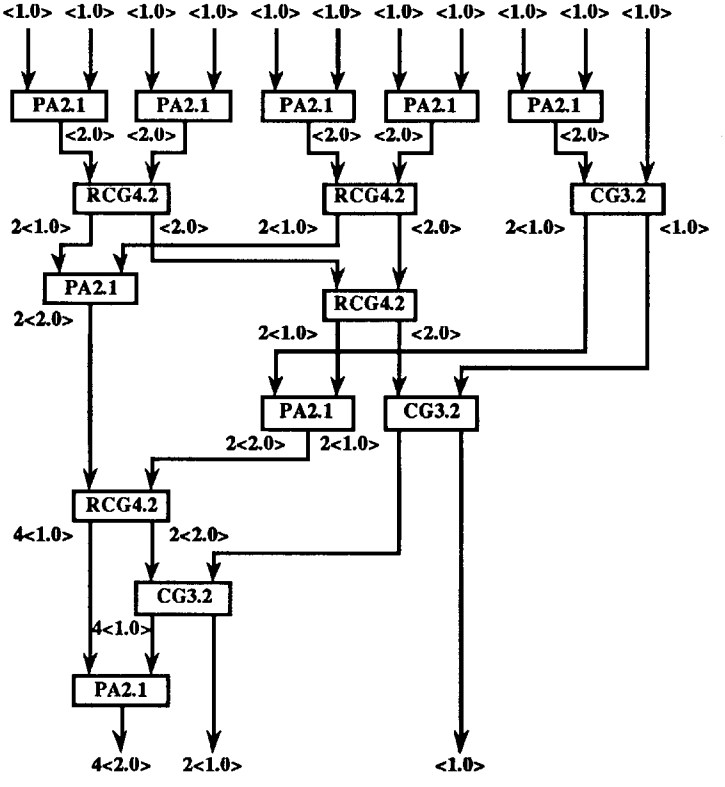

Figure 1: Design of a Radix-2, 11-Bit Parallel Counter

equations represented in the information loss table of figure 1 is as follows.

$$
\begin{array}{lrcl}
1 & 4\langle 2^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle & \Leftarrow & \langle 2^0\rangle + \langle 2^0\rangle + \langle 2^0\rangle + \langle 2^0\rangle + \langle 2^0\rangle + \langle 1^0\rangle \\
2 & 4\langle 2^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle & \Leftarrow & 2\langle 1^0\rangle + 2\langle 1^0\rangle + 2\langle 1^0\rangle + \langle 2^0\rangle + \langle 2^0\rangle + \langle 1^0\rangle \\
3 & 4\langle 2^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle & \Leftarrow & 2\langle 2^0\rangle + 2\langle 1^0\rangle + 2\langle 1^0\rangle + \langle 2^0\rangle + \langle 1^0\rangle \\
4 & 4\langle 2^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle & \Leftarrow & 2\langle 2^0\rangle + 2\langle 2^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle \\
5 & 4\langle 2^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle & \Leftarrow & 4\langle 1^0\rangle + 2\langle 2^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle \\
6 & 4\langle 2^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle & \Leftarrow & 4\langle 1^0\rangle + 4\langle 1^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle \\
7 & 4\langle 2^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle & \Leftarrow & 4\langle 2^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle
\end{array}
$$

Figure 2 shows the information loss table and block diagram for the radix-3 parallel counter specified by the following decomposition equation.

$$
\begin{gathered}
9\langle 1^0\rangle + 3\langle 2^0\rangle + \langle 2^0\rangle \Leftarrow \\
\langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle + \langle 1^0\rangle
\end{gathered}
$$

In this case, $\delta_{in} = 11$ and $\delta_{out} = 17$, so the equation represents a structure which is realizable but not decomposable. We must add mythical inputs for which $\delta = 6$ and $\omega = 0$. We do this by adding $3\langle 2^0\rangle = 0$.

4.7 A Carry Save Adder

Figure 3 shows the information loss table and block diagram for a 3-bit radix-2 unsigned carry save adder as specified by the equation

$$
\begin{gathered}
8\langle 1^0\rangle + 4\langle 2^0\rangle + 2\langle 2^0\rangle + \langle 2^0\rangle \Leftarrow \\
(4\langle 2^0\rangle + 2\langle 2^0\rangle + \langle 2^0\rangle) + (4\langle 1^0\rangle + 2\langle 1^0\rangle + \langle 1^0\rangle) + \langle 1^0\rangle\,.
\end{gathered}
$$

The first subexpression on the right hand side corresponds to the accumulating carry save sum. The output of a bit position in a carry save adder is a digit set that can represent values between 0 and 2 which is the $\langle 2^0\rangle$ digit set. The second subexpression represents the unsigned radix-2 number that is to be added to the accumulating carry save sum. The third subexpression is the carry in. Here, $\delta_{out} = \delta_{in} = 22$ so no mythical inputs are needed.

4.8 A Radix-4 Signed Digit Adder

Signed digit adders are similar to carry save adders, except that both inputs are in redundant form. Consider the design of a radix-4 structure that uses $\langle 4^2\rangle$ digits. It is a three level structure requiring three decomposition equations.

$$
\begin{array}{lrcl}
1 & 4\langle 1^1\rangle + 2\langle 1^0\rangle + \langle 2^0\rangle & \Leftarrow & (2\langle 1^1\rangle + \langle 2^0\rangle) + (2\langle 1^1\rangle + \langle 2^0\rangle) \\
2 & 4\langle 1^0\rangle + 2\langle 1^1\rangle + \langle 1^0\rangle & \Leftarrow & (2\langle 1^0\rangle + \langle 2^0\rangle) + \langle 1^1\rangle \\
3 & 2\langle 1^1\rangle + \langle 2^0\rangle & \Leftarrow & (2\langle 1^1\rangle + \langle 1^0\rangle) + \langle 1^0\rangle
\end{array}
$$

Lev	Operator (#,W) Name	λ	9 $\langle 1^0 \rangle$	3 $\langle 4^0 \rangle \langle 3^0 \rangle \langle 2^0 \rangle \langle 1^0 \rangle$	1 $\langle 3^0 \rangle \langle 2^0 \rangle \langle 1^0 \rangle$
I		11			11
M				1	
0		13		1	11
1	(4,1) $PA_{2.1}(000)$	13		1	4 3
2	(2,1) $PA_{3.2}(000)$	11		1	2 2 1
3	(2,1) $CG_{5.3}(0000)$	9		1 2	2 1
4	(1,3) $PA_{3.2}(000)$				
	(1,1) $PA_{3.2}(000)$	7		1 1	1 1
5	(1,3) $PA_{4.3}(000)$				
	(1,1) $CG_{5.3}(0000)$	6		1 1	1
6	(1,3) $CG_{5.4}(0000)$	5	1	1	1

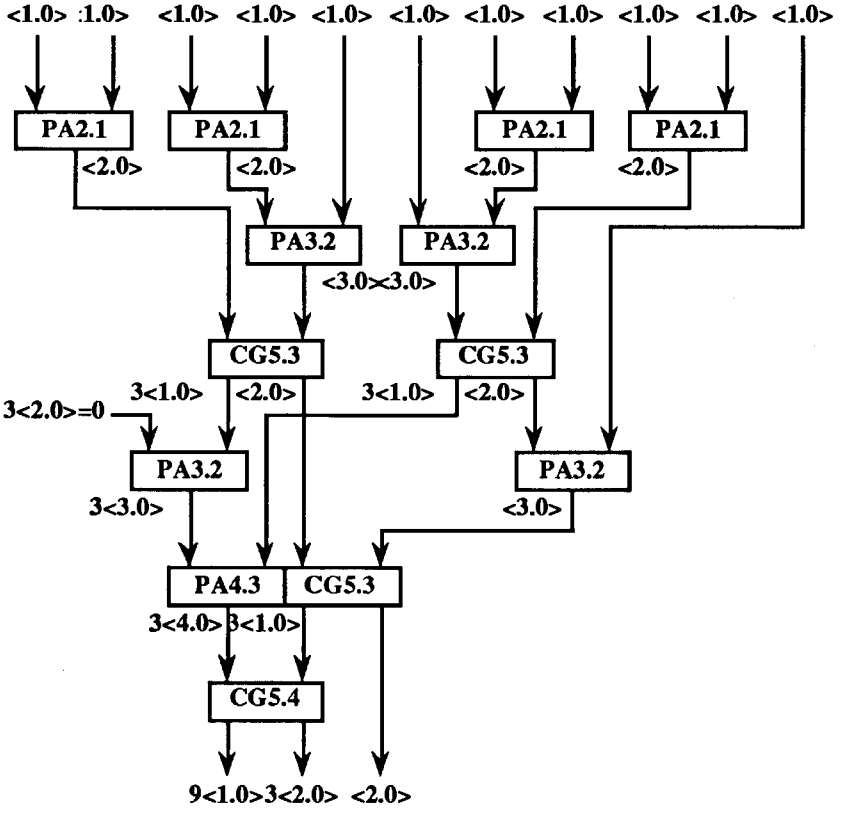

Figure 2: Design for a Radix-3, 11-Bit Parallel Counter

Lev	Operator (#,W) Name	λ	8 $\langle 1^0 \rangle$	4 $\langle 2^0 \rangle \langle 1^0 \rangle$	2 $\langle 2^0 \rangle \langle 1^0 \rangle$	1 $\langle 2^0 \rangle \langle 1^0 \rangle$
I				1 1	1 1	1 2
M						
0		10		1 1	1 1	1 2
1	(1,4) $CG_{3.2}(0000)$					
	(1,2) $CG_{3.2}(0000)$					
	(1,1) $CG_{3.2}(0000)$	7	1	2	2	2
2	(1,4) $PA_{2.1}(000)$					
	(1,2) $PA_{2.1}(000)$					
	(1,1) $PA_{2.1}(000)$	7	1	1	1	1

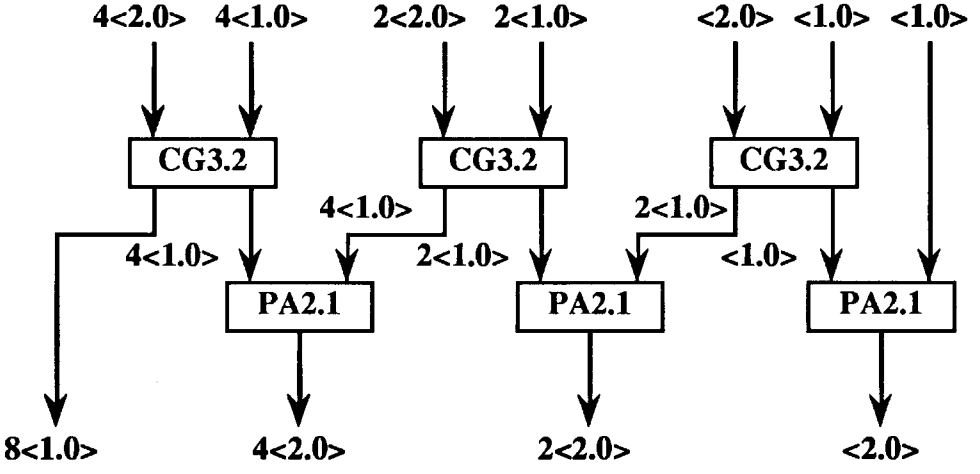

Figure 3: Design of a 3-Bit Carry Save Unsigned Adder

Only the second module requires a mythical input. For this module, $\delta_{out} = 7$, $\omega_{out} = 2$, $\delta_{in} = 5$ and $\omega_{in} = 1$ so it requires a $\langle 1^1 \rangle$ mythical input at weight 2. Decomposition results in the information loss tables and block diagram in figure 4. Note that, depending on its logic implementation, the $PA_{2.1}(110)$ that uses the mythical input may possibly be optimized away.

4.9 A Radix-2 Redundant Array Multiplier

Array (or parallel) multipliers have been discussed at length in the literature (e.g., [13, 14, 26, 28]). A radix-2 n-digit redundant array multiplier is one in which the starting representation is redundant as in

$$2^{n-1}\langle 2^1 \rangle + \cdots + 2^0 \langle 2^1 \rangle .$$

Array multipliers are essentially an array of *elementary multipliers* followed by a summation network. Consider a small, two-bit array multiplier. In this case we would need an array of four elementary multipliers that each perform

$$\langle 2^1 \rangle = \langle 2^1 \rangle \cdot \langle 2^1 \rangle$$

where the *set multiplication* operation is analagous to set addition. That is,

$$D_0 \cdot D_1 = \{(d_0 \cdot d_1) \mid d_0 \in D_0 \text{ and } d_1 \in D_1\}.$$

Set multiplication is associative and commutative and is also distributive over set addition.. Although in this particular elementary multiplier the result is a digit set, set multiplication applied to digit sets does not necessarily produce a digit set.

The overall structure of the elementary multiplier can be summarized as the following equation:

$$4\langle 2^1 \rangle + 2\langle 2^1 \rangle + 2\langle 2^1 \rangle + \langle 2^1 \rangle \leftarrow \left(2\langle 2^1 \rangle + \langle 2^1 \rangle\right) \cdot \left(2\langle 2^1 \rangle + \langle 2^1 \rangle\right).$$

The input does not represent a digit set since the multiplication $\langle 6^3 \rangle \cdot \langle 6^3 \rangle$ is

$$\{\overline{3},\overline{2},\overline{1},0,1,2,3\} \cdot \{\overline{3},\overline{2},\overline{1},0,1,2,3\} \equiv \{\overline{9},\overline{6},\overline{4},\overline{3},\overline{2},\overline{1},0,1,2,3,4,6,9\}$$

and the values 5, 7, and 8 and $\overline{5}$, $\overline{7}$ and $\overline{8}$ are not present. It can, however, be represented as a $\langle 30^{15} \rangle$ digit set even though many values are missing. Figure 5 shows the information loss table for the summation network of the multiplier and the block diagram for the entire multiplier. The output of the elementary multiplier structure is the input to a summation network described by the decomposition equation

$$8\langle 2^1 \rangle + 4\langle 2^1 \rangle + 2\langle 2^1 \rangle + \langle 2^1 \rangle \Leftarrow 4\langle 2^1 \rangle + 2\langle 2^1 \rangle + 2\langle 2^1 \rangle + \langle 2^1 \rangle$$

Lev	Operator (#,W) Name	λ	4 $\langle 1^1\rangle\langle 1^0\rangle$	2 $\langle 2^2\rangle\langle 1^1\rangle\langle 1^0\rangle$	1 $\langle 2^0\rangle\langle 1^1\rangle\langle 1^0\rangle$
I		6		2	2
M					
0		6		2	2
1	(1,2) $PA_{2.1}(211)$				
	(1,1) $RCG_{4.2}(0000)$	5		1 1	1
2	(1,2) $CG_{3.2}(1020)$	4	1	1	1

Lev	Operator (#,W) Name	λ	4 $\langle 1^1\rangle\langle 1^0\rangle$	2 $\langle 2^1\rangle\langle 1^1\rangle\langle 1^0\rangle$	1 $\langle 2^0\rangle\langle 1^1\rangle\langle 1^0\rangle$
I		4		1	1 1
M				1	
0		5		1 1	1 1
1	(1,2) $PA_{2.1}(110)$				
	(1,1) $CG_{3.2}(0101)$	4		1 1	1
2	(1,2) $CG_{3.2}(1011)$	3	1	1	1

Lev	Operator (#,W) Name	λ	2 $\langle 1^1\rangle$	1 $\langle 2^0\rangle\langle 1^0\rangle$
I		3	1	2
M				
0		3	1	2
1	(1,2) $PA_{2.1}(000)$	3	1	1

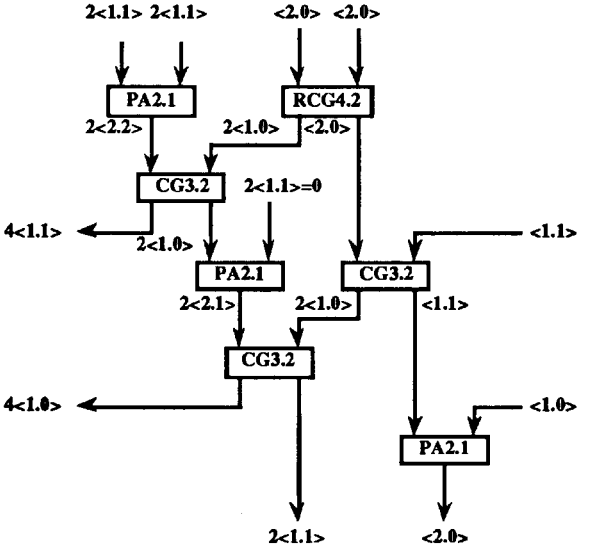

Figure 4: Design of a Radix-4 Signed Digit Adder

Table 5: Binary Information Contents of Decomposition Operators

Operator	α_{in}	α_{out}	β_{in}	β_{out}	λ_{in}	λ_{out}	$\triangle\beta$	$\triangle\alpha$	Var.
$\langle 1^* \rangle + \langle 1^* \rangle \Leftarrow \langle 2^* \rangle$	0	2	1	0	2	2	1	0	z
$\langle 2^* \rangle \Leftarrow \langle 1^* \rangle + \langle 1^* \rangle$	2	0	0	1	2	2	−1	0	y
$2\langle 1^* \rangle + \langle 1^* \rangle \Leftarrow \langle 2^* \rangle + \langle 1^* \rangle$	1	2	1	0	3	2	1	1	x
$2\langle 1^* \rangle + \langle 2^* \rangle \Leftarrow \langle 2^* \rangle + \langle 2^* \rangle$	0	1	2	1	4	3	1	1	w
$r\langle 1^* \rangle + \langle (r-1)^* \rangle \Leftarrow 2\langle (r-1)^* \rangle + \langle 1^* \rangle$	1	1					0	0	V_r

in which $\delta_{out} = 30$, $\omega_{out} = 15$, $\delta_{in} = 16$ and $\omega_{in} = 8$. Therefore a mythical input with $\delta = 12$ and $\omega = 6$ is required. The mythical input chosen is

$$4\left\langle 2^1 \right\rangle + 2\left\langle 2^1 \right\rangle .$$

5 Cost Estimation for Decomposition

With some limitations, it is possible to estimate the cost of a structure in terms of the number of decomposition operators in advance of detailed design. First, we list the number α of $\langle 1^* \rangle$ digit sets, the number β of $\langle 2^* \rangle$ digit sets, and the information content λ for the inputs and outputs of each of the binary decomposition operators as shown in table 5. The net changes $\triangle\beta$ and $\triangle\lambda$ for each operator are then determined. $\triangle\alpha$ is not listed, since $\triangle\lambda = 2\triangle\beta + \triangle\alpha$, so only two of the parameters are independent.

The specifications of a structure designate the nature of its inputs and outputs, so that $\triangle\beta$ and $\triangle\lambda$ can be determined. The structure must be decomposable, so these calculations must include any mythical inputs. After decomposition, an unknown number y of partial adders, x carry generators, and w redundant carry generators will have been used (for binary decomposition $z = V_r = 0$). Multiplying the number of operators of each class by the corresponding $\triangle\beta$ and summing, we get

$$\begin{aligned} \triangle\beta &= (w + x) - y \\ \triangle\lambda &= (w + x) \end{aligned} .$$

Since $\triangle\beta$ and $\triangle\lambda$ are known, one can determine that

$$\begin{aligned} w + x &= \triangle\lambda \\ y &= \triangle\lambda - \triangle\beta \end{aligned} .$$

Thus, the total number of carry generators (including redundant carry generators), is equal to the information loss $\triangle\lambda$ of the structure and the number of partial adders is determined from the change in the number of $\langle 2^* \rangle$ digit sets.

Lev	Operator (#,W) Name	λ	8 $\langle 2^1\rangle\langle 1^1\rangle\langle 1^0\rangle$	4 $\langle 2^1\rangle\langle 2^0\rangle\langle 1^1\rangle\langle 1^0\rangle$	2 $\langle 2^1\rangle\langle 2^0\rangle$	1 $\langle 2^1\rangle$
I		8		1	2	1
M		4		1	1	
0		12		2	3	1
1	(1,4) $RCG_{4.2}(1011)$					
	(1,2) $RCG_{4.2}(1011)$	10	1	1 1	1 1	1
2	(1,4) $CG_{3.2}(0101)$					
	(1,2) $RCG_{4.2}(0101)$	8	1 1	1 1	1	1
3	(1,8) $PA_{2.1}(101)$					
	(1,4) $PA_{2.1}(101)$	8	1	1	1	1

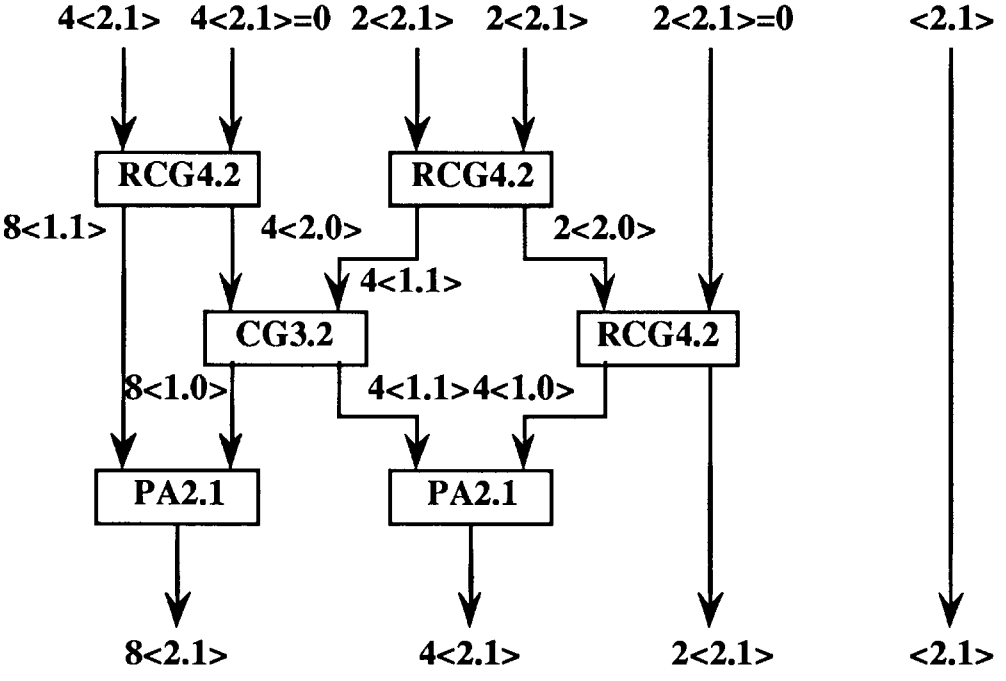

Figure 5: Design of a Two-Digit Radix-2 Redundant Array Multiplier

As an example, consider the 11-bit parallel counter of section 4.6. From its decomposition equations, $\alpha_{in} = 11$, $\beta_{in} = 0$, $\lambda_{in} = 11$, $\alpha_{out} = 2$, $\beta_{out} = 1$ and $\lambda_{out} = 4$; so $\Delta\lambda = 7$ and $\Delta\beta = -1$. Therefore the implementation will use $w + x = \Delta\lambda = 7$ (redundant) carry generators and $y = \Delta\lambda - \Delta\beta = 8$ partial adders. This agrees with the detailed design of figure 1.

The cost estimates for radix r are fairly simple if binary decomposition operators and a single radix r carry generator are used as discussed in section 4.1[1]. Although the exact expression for the radix r carry generator is not known unless r is known, note that the digit sets $\langle 1^{\star} \rangle$ and $\langle (r-1)^{\star} \rangle$ each appear once on each side of the expression. Therefore, independent of r, $\Delta\beta$ and $\Delta\lambda$ are 0. The value of V_r will be the diminished cardinality of the digit sets whose coefficients are r or binary multiples of r. For higher radices, with z and V_r no longer 0, the previous calculations become

$$\Delta\beta = (w + x) - (y - z)$$
$$\Delta\lambda = (w + x)$$

from which one determines that

$$w + x = \Delta\lambda$$
$$y - z = \Delta\lambda = \Delta\beta$$

and, as stated above, V_r is equal to the sum of the diminished cardinalities of the terms whose coefficients are $2^k r$ $(k = 0, 1, 2, \cdots)$.

As an example, consider a more or less conventional design of a decimal adder whose decomposition equation is

$$10\langle 1^0 \rangle + \langle 9^0 \rangle \Leftarrow \langle 9^0 \rangle + \langle 9^0 \rangle + \langle 1^0 \rangle$$

where the $\langle 1^0 \rangle$ digit sets are carries and the $\langle 9^0 \rangle$ digit sets are decimal digits. Using the binary representation of $\langle 9^0 \rangle$, the decomposition equation becomes

$$10\langle 1^0 \rangle + (4\langle 1^0 \rangle + 2\langle 2^0 \rangle + \langle 1^0 \rangle) \Leftarrow$$
$$(4\langle 1^0 \rangle + 2\langle 2^0 \rangle + \langle 1^0 \rangle) + (4\langle 1^0 \rangle + 2\langle 2^0 \rangle + \langle 1^0 \rangle) + \langle 1^0 \rangle .$$

From this expression, $\alpha_{in} = 5$, $\beta_{in} = 2$, $\lambda_{in} = 9$, $\alpha_{out} = 3$, $\beta_{out} = 1$, $\lambda_{out} = 5$, $\Delta\lambda = 4$ and $\Delta\beta = 1$. The cost of this decimal adder is therefore

Binary (Redundant) Carry Generators	$w + x = \Delta\lambda = 4$
Partial Adders	$y - z = \Delta\lambda - \Delta\beta = 3$
Radix-5 Carry Generators	$V_r = 1$.

Since the three digit $\langle 1^0 \rangle$ digit sets of weight 1 will generate a $\langle 1^0 \rangle$ carry digit set of weight 2, it is possible to predict that, in this case, $z = 0$. The information loss table for this adder is shown in figure 6.

[1] N.B. At radices 13,17,19,29 and higher primes, the analysis that follows does not apply. It must be augmented by considering the effects of the fanout function.

Lev	Operator (#,W) Name	λ	10 $\langle 1^0 \rangle$	8 $\langle 1^0 \rangle$	4 $\langle 2^0 \rangle \langle 1^0 \rangle$	2 $\langle 2^0 \rangle \langle 1^0 \rangle$	1 $\langle 2^0 \rangle \langle 1^0 \rangle$
I		9			2	2	3
M							
0		9			2	2	3
1	(1,4) $PA_{2.0}(000)$						
	(1,2) $RCG_{4.0}(0000)$						
	(1,1) $PA_{2.0}(000)$	8			1 1	1	1 1
2	(1,4) $CG_{3.0}(0000)$						
	(1,1) $CG_{3.0}(0000)$	6		1	1	1 1	1
3	(1,2) $CG_{3.0}(0000)$	5		1	2	1	1
4	(1,4) $PA_{2.0}(000)$	5		1	1	1	1
5	(1,2) $CG_{9.0}(0000)$	4	1		1	1	1

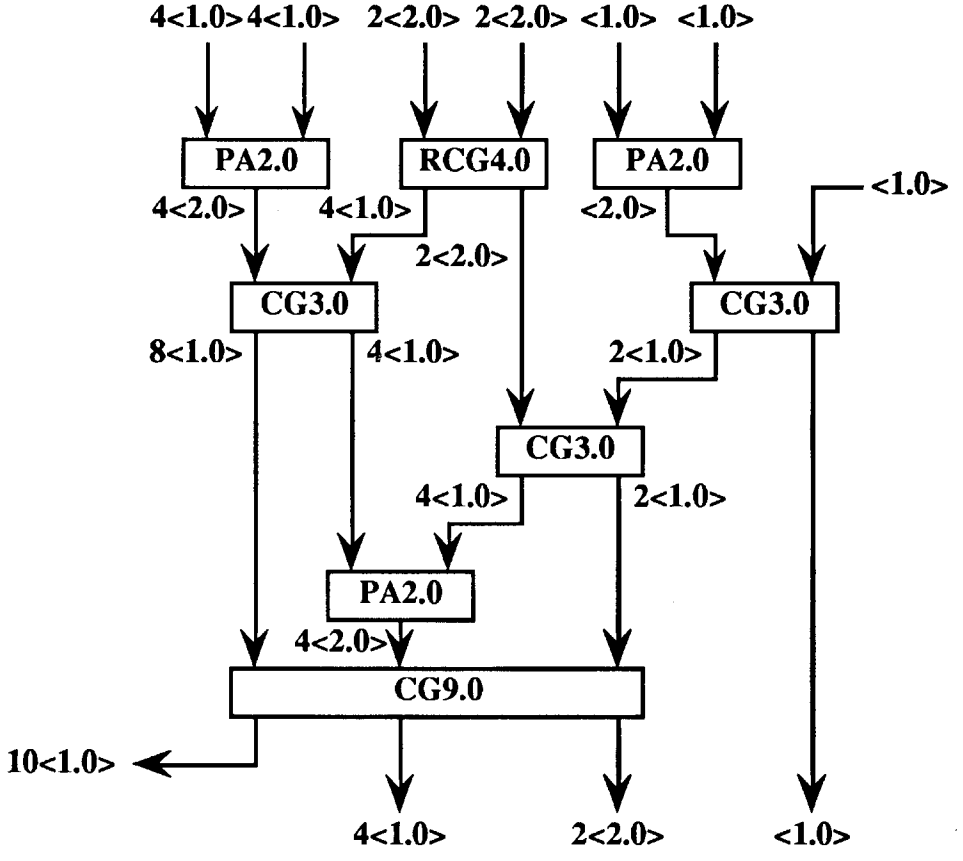

Figure 6: Design of a Radix-10 Adder

6 Logic Design

Once the set of classes of decomposition operators has been defined based on the application, a logic design must be completed for each decomposition operator. Once the logic design for each operator is completed, a circuit module implementing that operator can be readily designed. Logic design for decomposition operators, while not difficult, is not as straightforward as for most combinational logic circuits. There are two significant problems:

- selecting an implementation format for a digit set, and
- dealing with *coupled don't cares* in the logical design.

The following sections discuss, albeit cursorily, these two problems. If optimization for speed and design complexity is not desired, then the problem of logical design is considerably simplified.

6.1 Implementation Formats for Digit Sets

For each allowable digit set (in each of its offset variations), one must first select a mapping of its values to a set of binary variables. Such a mapping is called a *format.* For the binary digit sets, $\langle 1^* \rangle$, the implementation requires only a single binary variable and so the mapping is trivial. There are two possible ways to do the mapping, but the mappings are equivalent under negation of the binary variable, so there is only one distinct format as shown in table **6**. Generally the non-negated mapping is used since an arithmetic zero maps to a logical zero.

For digit sets of higher diminished cardinality, the situation is significantly more complex. Let us consider the ternary digit sets, $\langle 2^* \rangle$, which have three values that map onto the four states of two binary variables. This mapping may be done in $4!/(4-3)! = 24$ ways. The fourth state of the two binary variables may map to any of the three values or none of them, effectively multiplying the number of possible mappings by 4. Finding the best state assignment from among these 96 is a formidable task unless some simplifications can be made.

One simplification occurs by recognizing that under permutation and negation of the two binary variables, the 24 main mappings can be separated into three equivalence classes of 8 mappings each (four due to the four possible negations and four due to two permutations). Since there are four possible interpretations of the extra state, there are four possible interpretations for each of the three equivalence classes for a total of 12. All other possible implementations can be generated easily from these 12 by simply permuting the order of the binary variables and/or by negating one or both.

The second optimization occurs when assigning a specific value to the unmapped state. In these cases, one of the values is duplicated and it does not matter which

copy of the value is mapped to which state of the two binary variables. There are three of these duplicate cases which can be eliminated from the 12 cases, leaving 9 distinct *formats*; only nine distinct logical designs need to be done in order to easily generate all 96 possible logical designs. From these, the best can be selected for physical implementation. Table 7 presents the formats for the ternary digit sets. The mapping used in each format may be any of the eight in the equivalence class.

For radix 3, there are only 24 distinct mappings of the four digit set values to the four states of two binary variables. Again, under permutation and negation of the binary variables, there are three equivalence classes that represent all 24 possible logical designs. So, three distinct logical designs permit easy generation of the 24 possible logical designs. Table 8 shows the 3 formats for the quaternary digit sets. For higher valued digit sets, selecting an optimal format becomes practically impossible. The number of formats is given by

$$\mathcal{F} = \frac{M}{V} \cdot D - cS$$

where M is the number of possible mappings

$$M = \frac{\left(2^{\lceil \log_2(\delta+1) \rceil}\right)!}{\left(2^{\lceil \log_2(\delta+1) \rceil} - (\delta+1)\right)!}$$

V is the reduction factors due to permuting and negating the boolean variables

$$V = \lceil \log_2(\delta+1) \rceil) \cdot 2^{\lceil \log_2(\delta+1) \rceil)}$$

D is the number of additional formats introduced through the use of don't cares

$$D = \frac{\left(2^{\lceil \log_2(\delta+1) \rceil}\right)!}{(\delta+1)!}$$

and S is the number of redundant mappings due to don't cares (i.e., when there are two or more copies of an element of a digit set in a mapping)

$$S = (\delta+1)^{2^{\lceil \log_2(\delta+1) \rceil} - (\delta+1)}$$

but S is only used when there are redundant mappings, so c is

$$c = \left\{ \begin{array}{l} 0 \text{ when } (\delta+1) = 2^{\lceil \log_2(\delta+1) \rceil} \\ 1 \text{ when } (\delta+1) < 2^{\lceil \log_2(\delta+1) \rceil} \end{array} \right\}$$

Table 6: Format for the Binary Digit Sets

Var.	**Format 1**	
a	$\langle 1^0 \rangle$	$\langle 1^1 \rangle$
0	0	0
1	1	$\overline{1}$

Table 7: Formats for the Ternary Digit Sets

Var.	**Format 1**			**Format 2**			**Format 3**		
$\mathbf{b_g\ b_e}$	$\langle 2^2 \rangle$	$\langle 2^1 \rangle$	$\langle 2^0 \rangle$	$\langle 2^2 \rangle$	$\langle 2^1 \rangle$	$\langle 2^0 \rangle$	$\langle 2^2 \rangle$	$\langle 2^1 \rangle$	$\langle 2^0 \rangle$
00	0	0	0	0	0	0	0	0	0
01	$\overline{1}$	1	1	$\overline{1}$	1	1	—	$\overline{1}$	—
10	$\overline{1}$	$\overline{1}$	1	$\overline{2}$	—	2	$\overline{2}$	—	2
11	$\overline{2}$	0	2	—	$\overline{1}$	—	$\overline{1}$	1	1

Var.	**Format 4**			**Format 5**			**Format 6**		
$\mathbf{b_g\ b_e}$	$\langle 2^2 \rangle$	$\langle 2^1 \rangle$	$\langle 2^0 \rangle$	$\langle 2^2 \rangle$	$\langle 2^1 \rangle$	$\langle 2^0 \rangle$	$\langle 2^2 \rangle$	$\langle 2^1 \rangle$	$\langle 2^0 \rangle$
00	0	0	0	0	0	0	0	0	0
01	$\overline{1}$	1	1	—	1	—	0	1	0
10	$\overline{2}$	0	2	$\overline{1}$	$\overline{1}$	1	$\overline{1}$	$\overline{1}$	1
11	$\overline{1}$	$\overline{1}$	1	$\overline{2}$	—	2	$\overline{2}$	$\overline{1}$	2

Var.	**Format 7**			**Format 8**			**Format 9**		
$\mathbf{b_g\ b_e}$	$\langle 2^2 \rangle$	$\langle 2^1 \rangle$	$\langle 2^0 \rangle$	$\langle 2^2 \rangle$	$\langle 2^1 \rangle$	$\langle 2^0 \rangle$	$\langle 2^2 \rangle$	$\langle 2^1 \rangle$	$\langle 2^0 \rangle$
00	0	0	0	0	0	0	0	0	0
01	$\overline{1}$	$\overline{1}$	1	$\overline{2}$	1	2	$\overline{1}$	$\overline{1}$	1
10	$\overline{2}$	1	2	$\overline{2}$	1	2	$\overline{2}$	$\overline{1}$	2
11	$\overline{2}$	1	2	$\overline{1}$	$\overline{1}$	1	0	1	0

Table 8: Formats for the Quaternary Digit Sets

Var.	**Format 1**				**Format 2**				**Format 3**			
$\mathbf{c_g\ c_e}$	$\langle 3^3 \rangle$	$\langle 3^2 \rangle$	$\langle 3^1 \rangle$	$\langle 3^0 \rangle$	$\langle 3^3 \rangle$	$\langle 3^2 \rangle$	$\langle 3^1 \rangle$	$\langle 3^0 \rangle$	$\langle 3^3 \rangle$	$\langle 3^2 \rangle$	$\langle 3^1 \rangle$	$\langle 3^0 \rangle$
00	0	0	0	0	0	0	0	0	0	0	0	0
01	$\overline{1}$	$\overline{1}$	1	1	$\overline{1}$	$\overline{1}$	1	1	$\overline{3}$	1	$\overline{1}$	3
10	$\overline{2}$	1	$\overline{1}$	2	$\overline{3}$	$\overline{2}$	2	3	$\overline{2}$	$\overline{2}$	2	2
11	$\overline{3}$	$\overline{2}$	2	3	$\overline{2}$	1	$\overline{1}$	2	$\overline{1}$	$\overline{1}$	1	1

The total number of formats is given by the equation

$$\mathcal{F} = \frac{\left(2^{\lceil \log_2(\delta+1) \rceil}\right) \cdot \left[\left(2^{\lceil \log_2(\delta+1) \rceil} - 1\right)!\right]^2}{\left(2^{\lceil \log_2(\delta+1) \rceil} - (\delta+1)\right)! \cdot \lceil \log_2(\delta+1) \rceil ! \cdot (\delta+1)!} - c \cdot (\delta+1)^{2^{\lceil \log_2(\delta+1) \rceil} - (\delta+1)}.$$

So, for example, using the 8 states of 3 bits, the $\langle 6^{\star} \rangle$ digit sets can be implemented using 6713 different formats.

6.2 Logical Design of Decomposition Operators

Once an implementation format has been selected, the task of performing the logical design of the decomposition operators can begin. Selection of the optimal format in general requires designing the complete set of decomposition operators in each format and comparing the results. This is simply tedious, not difficult as we now show. We illustrate the logic design process by considering the design of the $\mathrm{RCG}_{4.2}(0000) = 2\langle 1^0 \rangle + \langle 2^0 \rangle \Leftarrow \langle 2^0 \rangle + \langle 2^0 \rangle$ operator. Let us select format 2 for the $\langle 2^0 \rangle$ digit set. First we generate the truth table in figure 7.

Note that the entries in the output side of the table for the arithmetic value of 2 has a pair of value pairs from which to select during the design. These are known as *coupled don't cares*. In effect, they link the minimizations of the logical designs of two or more outputs from a multiple-output function. In the case of the arithmetic value of two, we can represent the output as either 100 or 010. It does not matter which one we pick to represent each row in the truth table, but we must pick one of them. Representing coupled don't cares in a Karnaugh map by a k, we arrive at the set of three maps in figure 7.

The k' in the second map indicate that when we pick a boolean value at a location in the first map, we must pick its inverse in the same location in the second map. With the three coupled don't cares, there are eight possible logical implementations from which we must choose the best. This occurs when $k_{0010} = 1$, $k_{0101} = 0$ and $k_{1000} = 1$. In this case, the corresponding minimized logical equations are:

$$\begin{aligned} 2a^0 &= b^0_{g0} + b^0_{e0} \\ b^0_{go} &= b^0_{g0} \cdot b^0_{g1} + b^0_{e0} \cdot b^0_{e1} \\ b^0_{eo} &= b^0_{e0} \oplus b^0_{e1} \end{aligned}$$

The problem with coupled don't cares arises when there is redundancy in the output of an arithmetic set expression, that is, when there is more than one way to represent an arithmetic value. This is avoided in the partial adders and the carry generators, but the redundant carry generators always involve design with coupled don't cares. The design of elementary multipliers often involves coupled don't cares, particularly at higher radices.

Inputs				**Arithmetic Value**	**Outputs**		
$\langle 2^0 \rangle$		$\langle 2^0 \rangle$			$2\langle 1^0 \rangle$	$\langle 2^0 \rangle$	
$\mathbf{b}^0_{g0}$	$\mathbf{b}^0_{e0}$	$\mathbf{b}^0_{g1}$	$\mathbf{b}^0_{e1}$		$\mathbf{2a}^0$	$\mathbf{b}^0_{go}$	$\mathbf{b}^0_{eo}$
0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	1
0	0	1	0	2	01 / 10		0
0	0	1	1	—	—	—	—
0	1	0	0	1	0	0	1
0	1	0	1	2	01 / 10		0
0	1	1	0	3	1	0	1
0	1	1	1	—	—	—	—
1	0	0	0	2	01 / 10		0
1	0	0	1	3	1	0	1
1	0	1	0	4	1	1	0
1	0	1	1	—	—	—	—
1	1	0	0	—	—	—	—
1	1	0	1	—	—	—	—
1	1	1	0	—	—	—	—
1	1	1	1	—	—	—	—

$2a^0$

	00	01	11	10
00	0	0	–	k
01	0	k	–	1
11	–	–	–	–
10	k	1	–	1

b^0_{go}

	00	01	11	10
00	0	0	–	k'
01	0	k'	–	0
11	–	–	–	–
10	k'	0	–	1

b^0_{eo}

	00	01	11	10
00	0	1	–	0
01	1	0	–	1
11	–	–	–	–
10	0	1	–	0

Figure 7: Format 2 Truth Table and Karnaugh Maps for $RCG_{4.2}(0000)$

7 Conclusions

The Set Theory of Arithmetic Decomposition forms a basis for designing and evaluating the designs of arithmetic units at any integral radix. It ties together several known implementations of arithmetic units developed for radix 2. Decomposition allows the design of arithmetic circuits that were previously difficult or practically impossible to design. It facilitates the design of higher radix arithmetic units and has been used in a radix-16 variable precision arithmetic processor based on a signed digit number representation [6, 7, 8, 11, 12, 24]. The decomposition of the entire arithmetic unit took only a few minutes and, given the decomposition operators implemented as integrated circuit modules [4, 5], the layout of the arithmetic circuit took less than 4 days.

Diminished radix (one's) complement adders have almost entirely fallen into relative disfavor among systems implementors, as have sign magnitude adders. It is interesting that the Set Theory of Arithmetic Decomposition will not work for diminished radix complement or sign magnitude. This supports the general practice of dealing primarily with radix complement numbers. While the signed digit number representation is not prevalent in adders, it is used extensively in SRT division algorithms. The theory presented in this paper can be very useful in evaluating the implementations of high radix dividers [9].

The decomposition algorithm has been automated for the radix-2 redundant decomposition operators. We have also used full adders as decomposition operators and implemented them using partial adders and carry generators, so the addition of hierarchical decomposition operators to the process has been demonstrated. It is possible to extend decomposition to design using significantly more complex decomposition operators. Certainly a radix-16 full adder could be implemented as a block carry lookahead module and the decomposition algorithm could be used in conjunction with it to build larger block carry lookahead circuits.

Although limited to arithmetic circuits, the Set Theory of Arithmetic Decomposition represents an efficient method for designing very complex multiple-level combinational arithmetic circuits. The complexity of design using the theory is linear with the number of inputs whereas design using the more powerful Boolean algebra is exponential with the number of inputs. The decomposition algorithm generates a multiple output function using minimal multiple-level logic whereas other combinational logic minimization techniques are generally two-level and limited to a single function output.

The results of a decomposition have a one-to-one mapping with circuit structures for the multiple-level combinational logic function implementation, including internal connectivity. This is a significant advantage since these results can be used to directly and easily generate a circuit implementation.

References

1. Avizienis, A. "Signed-digit number representations for fast parallel arithmetic". *IRE Transactions on Electronic Computers*, EC-10, 9 (Sep. 1961) 389–400.

2. Borovec, R. T. *The Logical Design of a Class of Limited Carry-Borrow Propagation Adders.* Master's thesis, Univ. of Illinois at Urbana-Champaign (Aug. 1968).

3. Bratun, J. M. *et al.* "Multiply/divide unit for a high-performance digital computer". *IBM Technical Disclosure Bulletin*, 14, 6 (Nov. 1971) 1813–1816.

4. Carter, T. M. *Structured Arithmetic Tiling of Integrated Circuits.* PhD thesis, Dept. of Computer Science (Dec. 1983).

5. Carter, T. M. "Structured arithmetic tiling of integrated circuits". In *Proc. of the* 8^{th} *Symposium on Computer Arithmetic*, Como, ITALY (May 1987) 41–48.

6. Carter, T. M. *Cascade: A Hardware Alternative to Bignums.* Tech. Report UUCS-89-006, Univ. of Utah, Dept. of Computer Science (Apr. 1989).

7. Carter, T. M. *Cascade: Hardware for High/Variable Precision Arithmetic.* Tech. Report UUCS-89-005, Univ. of Utah, Dept. of Computer Science (Jan. 1989).

8. Carter, T. M. and Hollaar, L. A. "The implementation of a radix-16 digit-slice using a cellular vlsi technique". In *Proc. IEEE Int'l Conference on Computer Design*, Port Chester, NY (Nov. 1983) 688–691.

9. Carter, T. M. and Robertson, J. E. *Radix-16 Signed-Digit Division.* Tech. Report UUCS-88-004, Univ. of Utah, Dept. of Computer Science (Apr. 1988).

10. Chow, C. Y. and Robertson, J. E. "Logical design of a redundant binary adder". In *Proc. of the* 4^{th} *Symposium on Computer Arithmetic*, Santa Monica, CA (Oct. 1978) 25–27.

11. Chow, C. Y. F. *A Variable Precision Processor Module.* PhD thesis, Dept. of Computer Science (1980).

12. Chow, C. Y. F. "A variable precision processor module". In *Proc. IEEE Int'l Conference on Computer Design*, Port Chester, NY (Nov. 1983) 692–695.

13. Dadda, L.i. "Some schemes for parallel multipliers". *Alta Freg.*, 34 (1965) 349–356.

14. Dadda, L. "On parallel digital multipliers". *Alta Freg.*, 45 (1976) 574–580.

15. Foster, C. C. and Stockton, F. D. "Counting responders in an associative memory". *IEEE Transactions on Computers*, C-20 (1971) 1580–1583.

16. Hwang, K. *Computer Arithmetic: Principles, Architecture, and Design.* John Wiley and Sons, New York (1979).

17. MacSorley, O. L. "High-speed arithmetic in binary computers". *Proc. of the IRE*, 49, 1 (Jan. 1961) 67–91.

18. Nguyen, D. D. *A Symmetric Approach to the Design of Structures for Addition and Subtraction.* PhD thesis, Univ. of Illinois at Urbana-Champaign (Dec. 1980).

19. Ozarka, R. M. *The Design of Maximally Redundant Radix Four Arithmetic Structures.* Master's thesis, Univ. of Illinois at Urbana-Champaign (May 1980).

20. Pezaris, S. D'. "A 40^{ns} 17-bit-by-17-bit array multiplier". *IEEE Transactions on Computers*, C-20, 4 (Apr. 1971) 442–447.

21. Robertson, J. E. *A Deterministic Procedure for the Design of Carry-Save Adders and Borrow-Save Adders.* Report 235, Univ. of Illinois, Dept. of Computer Science (July 1967).

22. Robertson, J. E. "Parallel digital arithmetic unit utilizing a signed-digit format". (1969). U.S. Patent Number 3,462,589.

23. Robertson, J. E. "A systematic approach to the design of structures for arithmetic". In *Proc. of the* 5^{th} *Symposium on Computer Arithmetic* (May 1981) 35–41.

24. Robertson, J. E. "Design of the combinational logic for a radix-16 digit-slice for a variable precision processor module". In *Proc. IEEE Int'l Conference on Computer Design*, Port Chester, NY (Nov. 1983) 696–699.

25. Rohatsch, F. A. *A Study of Transformations Applicable to the Development of Limited Carry-Borrow Propagation Adders.* PhD thesis, Univ. of Illinois at Urbana-Champaign (June 1967).

26. Stenzel, W. *et al.* "A compact high speed parallel multiplication scheme". *IEEE Transactions on Computers*, C-26 (1973) 948–957.

27. Swartzlander, Earl E. "Parallel counters". *IEEE Transactions on Computers*, C-22 (1973) 1021–1024.

28. Wallace, C. S. "A suggestion for a fast multiplier". *IEEE Transactions on Electronic Computers*, EC-13, (Feb. 1964) 14–17.