



ELSEVIER

INTEGRATION, the VLSI journal 30 (2000) 13–53

INTEGRATION
theVLSI journal

www.elsevier.com/locate/vlsi

Redundant arithmetic, algorithms and implementations[☆]

Alejandro F. González, Pinaki Mazumder*

*The University of Michigan, Electrical Engineering and Computer Science Department, 1301 Beal Ave.,
Room 2215, Ann Arbor, MI 48109-2122, USA*

Received 9 August 2000

Abstract

Performance in many very-large-scale-integrated (VLSI) systems such as digital signal processing (DSP) chips, is predominantly determined by the speed of arithmetic modules like adders and multipliers. Even though redundant arithmetic algorithms produce significant improvements in performance through the elimination of carry propagation, efficient circuit implementations of these algorithms have been traditionally difficult to obtain. This work presents a survey of circuit implementations of redundant arithmetic algorithms. The described implementations are divided into three main groups: (1) conventional binary logic circuits, which encode the multivalued digits of redundant arithmetic into two or more binary digital signals; (2) current-mode multiple-valued logic circuits, which directly represent multivalued redundant digits using non-binary digital current signals; and (3) heterostructure and quantum electronic circuits, intended for very compact designs capable of operating at extremely high speeds. For each of the circuits, the operating principle is described and the main advantages and disadvantages of the approach are discussed and compared. © 2000 Published by Elsevier Science B.V.

Keywords: Redundant number systems; Redundant binary logic; Current-mode logic; Multiple-valued logic; Quantum electronic circuits

1. Introduction

Increasing the speed of arithmetic logic circuits is of extreme importance in the field of computing and signal processing circuits. On one hand, throughputs of the order of few gigaoperations per second are now conventional in large scientific vector processors and

[☆]This work was supported by the U.S. Army Research Office under the URI program, by the Defense Advanced Research Projects Agency, and by the National Science Foundation.

* Corresponding author.

E-mail addresses: algonz@eecs.umich.edu (A.F. González), mazum@eecs.umich.edu (P. Mazumder).

supercomputers [1]. This kind of performance is usually achieved by means of fast, densely integrated circuit technologies and highly parallel organizations. On the other hand, advanced signal processors with clock rates of the order of gigahertz are required for future system applications such as digital microwave receivers [1]. Speed in these processors is usually limited by the latency of their arithmetic units, especially that of multipliers.

Carry signals propagating through long chains of logic, as is the case in conventional ripple-carry adders, significantly hurt performance in arithmetic systems. In conventional adders, the worst-case latency is proportional to the number of digits involved in the operation, that is, the size of the operands. Moreover, multiplier circuits cascade a number of adders to sum the partial products. The importance of carry propagation effects is evident, considering that operand sizes tend to increase with the constant strive for greater processor power and precision. A standard for floating point arithmetic [2], for example, specifies 52-bit mantissas for the double-precision floating-point operands. Having addition latency proportional to 52 times the delay of an adder cell is obviously unacceptable, considering that the length of logic paths between latches is highly constrained in modern pipelined high-performance microprocessors.

Redundant number systems reduce or eliminate carry propagation chains in digital arithmetic circuits. In these number systems, redundancy is owed to the fact that, contrary to the case of conventional systems, redundant numbers can have more than one representation. For instance, in a radix-4 system with the digit set $\{-2, -1, 0, 1, 2\}$, numbers 0020, 0100, 10 $\bar{2}$ 0, and 1 $\bar{1}$ 00 all have arithmetic value 8. Redundancy allows addition algorithms in which carry propagation is completely eliminated. In these algorithms, a proper intermediate representation of the operand digit summation, $x_i + y_i$, is selected so that the final addition result can be generated using half-adders that do not require nor generate carry signals. More details on redundant arithmetic systems are given in Section 2.

Even though redundant addition techniques offer great improvements in computing performance, efficient circuit implementations of these algorithms have been traditionally difficult to achieve. A simple way to understand these difficulties is by noting the fact that digits in redundant systems are often not binary. This forces a departure from traditional binary logic circuits and conventional circuit techniques and technologies. In this work, we survey digital circuit implementations of redundant arithmetic algorithms. Conventional binary logic circuit techniques, described in Section 3, have been applied in redundant arithmetic systems by encoding redundant multi-valued digits into two or more binary signals. It is interesting to study the evolution of circuit implementations of redundant arithmetic systems and appreciate the different circuit techniques that have been inspired. Such circuit techniques often explore non-traditional areas of digital circuit design. Section 4 presents multiple-valued logic and current-mode circuits. These circuits are intended as a better match of the need for operation with non-binary digits. Heterostructure and quantum devices are used in alternative circuit techniques, described in Section 5, intended for very compact designs capable of operating at extremely high speeds.

2. Redundant number systems

This section presents the basic concepts of redundant arithmetic. Signed-digit arithmetic [3], a special case of redundant arithmetic, is described in more detail and is used to introduce the

properties that characterize redundant arithmetic systems. Such properties include, among others, the number format, the addition algorithm, the valid digit set, and the proper radix values. Other redundant number systems are addressed using Parhami’s unifying *generalized signed-digit* number representation [4].

2.1. Signed-digit number representations

Signed-digit systems were conceived with the purpose of implementing *totally parallel addition* [3], where carry propagation is eliminated. Carry propagation is eliminated by making each digit of the resulting sum a function of only two input digits. This is made possible by the redundancy of the number representation since a proper intermediate representation of the operand digit summation, $x_i + y_i$, is selected so that the final addition result can be generated using half-adders that do not require nor generate carry signals. The totally parallel addition algorithm can also be used to perform subtraction operations. The following paragraphs present the most important characteristics of signed-digit systems and the basic principles of the corresponding addition algorithm.

2.1.1. Properties of signed-digit number systems

The algebraic value of a signed-digit number is given by

$$Z = \sum_{i=-n}^m z_i r^{-i},$$

where r is a positive integer called the *radix*. In a redundant representation with radix r , each digit can assume more than r values, whereas in conventional number representations digits can assume exactly r values. The values of the radix and the number digits, z_i , should satisfy the condition of a unique representation for the algebraic value $Z = 0$. It is then easy to prove that the algebraic value Z is zero if, and only if, all digits of its signed-digit representation have the value $z_i = 0$. It is also evident that the sign of the algebraic value Z is determined by the sign of the most significant non-zero digit. Similarly, the signed-digit representation of $-Z$, the additive inverse of Z , is obtained by changing the sign of every non-zero z_i digit of Z .

Fig. 1 depicts the totally parallel addition approach in the signed-digit arithmetic system. The addition of two digits x_i and y_i is totally parallel if two conditions are satisfied. First, the sum digit

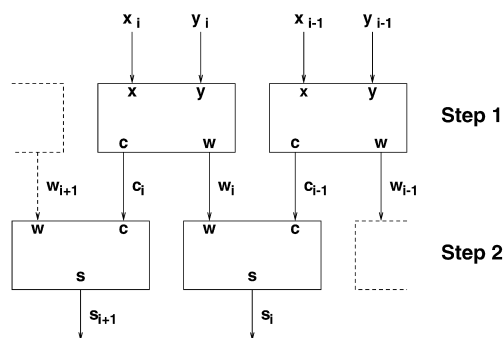


Fig. 1. Totally parallel addition approach in signed-digit number representation [3].

s_i is function only of the operand digits, x_i and y_i , and the carry digit c_{i-1} from the adjacent digit position. Second, the carry digit to the next position c_i is function only of the operand digits, x_i and y_i . Totally parallel subtraction $x_i - y_i$ is realized as the totally parallel addition of x_i and the additive inverse of y_i , that is, $x_i - y_i = x_i + (-y_i)$.

Totally parallel addition of two digits is performed in two steps, as depicted in Fig. 1. In the first step, a transfer digit output c_i and an interim sum output w_i are generated such that

$$x_i + y_i = rc_i + w_i. \quad (1)$$

In the second step, the final sum digit s_i is obtained as

$$s_i = w_i + c_{i-1}. \quad (2)$$

The required and allowed digit values for each of the variables involved in the two-step addition process can be derived from the definition of totally parallel addition and from the addition algorithm described by (1) and (2). The most important results of such derivation are as follows (for a complete analysis see [3]):

1. The smallest sufficient set of values for the carry digit is $c_i = \{-1, 0, 1\}$.
2. The upper bound for the magnitude of the interim sum is $|w_i| \leq r - 2$.
3. The lower bound for the radix value is $r > 2$.
4. For an odd radix, $r_o \geq 3$, the required (minimum) set of values for operand digits x_i and y_i consists of the sequence of $r_o + 2$ integers

$$\left\{ -\frac{1}{2}(r_o + 1), \dots, -1, 0, 1, \dots, \frac{1}{2}(r_o + 1) \right\}.$$

5. For an even radix, $r_e \geq 4$, the minimum set of values for operand digits x_i and y_i consists of the sequence of $r_e + 3$ integers

$$\left\{ -\left(\frac{1}{2}r_e + 1\right), \dots, -1, 0, 1, \dots, \frac{1}{2}r_e + 1 \right\}.$$

Minimum sets are the only allowed for radix-3 and radix-4 systems. For $r > 4$, however, there is more than one valid set of digit values. The sequence of integers

$$\{ -a, -(a-1), \dots, -1, 0, 1, \dots, a-1, a \}$$

meets the requirements for signed-digit number representations, where

$$\frac{1}{2}(r_o + 1) \leq a \leq r_o - 1 \quad \text{or} \quad \frac{1}{2}r_e + 1 \leq a \leq r_e - 1,$$

r_o is an odd integer $r_o \geq 3$, and r_e is an even integer $r_e \geq 4$. All signed-digit number representations can be described in terms of the allowed radix values and the allowed z_i digit values. The redundancy of a signed-digit system is said to be *minimal* when $a = 1/2(r_o + 1)$ or $a = 1/2r_e + 1$, and the redundancy is *maximal* when $a = r_o - 1$ or $a = r_e - 1$.

2.1.2. Signed-digit addition and subtraction

Two signed-digit numbers are added by means of the totally parallel addition algorithm described by (1) and (2). The rules for obtaining w_i , c_i , and s_i can be determined given the set of allowed values of w_i , w_{\min} and w_{\max} , as follows. From (1),

$$w_i = (x_i + y_i) - rc_i,$$

where

$$c_i = \begin{cases} 0 & \text{if } w_{\min} \leq x_i + y_i \leq w_{\max}, \\ 1 & \text{if } x_i + y_i > w_{\max}, \\ -1 & \text{if } x_i + y_i < w_{\min} \end{cases}$$

and

$$s_i = w_i + c_{i-1}.$$

Example 2.1. Signed-digit addition(Radix 10).

The allowed values for the digits are

$$w_i: \bar{5}, \bar{4}, \bar{3}, \bar{2}, \bar{1}, 0, 1, 2, 3, 4, 5,$$

$$c_i: \bar{1}, 0, 1;$$

$$s_i, x_i, y_i: \bar{6}, \bar{5}, \bar{4}, \bar{3}, \bar{2}, \bar{1}, 0, 1, 2, 3, 4, 5, 6.$$

Digits with negative values are identified with a bar above the integer. The addition operands are $x = 4.\bar{2}\bar{5}14\bar{3}$ (algebraic value $X = 3.75137$), and $y = \bar{2}.\bar{3}\bar{3}021$ (algebraic value $Y = -2.32979$). The addition procedure is as follows:

augend x:	4	.	$\bar{2}$	$\bar{5}$	1	4	$\bar{3}$
addend y:	$\bar{2}$.	$\bar{3}$	$\bar{3}$	0	2	1
Step (1):	0 + 2		0 + $\bar{5}$	$\bar{10}$ + 2	0 + 1	10 + $\bar{4}$	0 + $\bar{2}$
Step (2):	0		$\bar{1}$	0	1	0	
Sum s:	2	.	$\bar{6}$	2	2	$\bar{4}$	$\bar{2}$

The resulting sum is $s = 2.\bar{6}\bar{2}\bar{2}\bar{4}\bar{2}$, which as an algebraic value $S = 1.42158 = 3.75137 - 2.32979$.

2.2. Other redundant number systems

Stored-carry, stored-borrow, and the binary signed-digit (BSD) number systems are examples of other useful redundant arithmetic systems. In [4], Parhami proposed a *generalized signed-digit* number representation (GSD) where Avizienis' signed-digit system is named the *ordinary signed-digit* number system (OSD). In the generalized number system, the OSD and BSD systems are unified and other useful redundant number representations such as stored-carry and stored-borrow are included as special cases.

The generalized signed-digit number system is a positional system (a weight is associated with each digit position) with the digit set $\{-\alpha, -\alpha + 1, \dots, \beta - 1, \beta\}$, where $\alpha \geq 0$, $\beta \geq 0$, $\alpha + \beta + 1 > r$, and r is the radix of the number representation. The excluded case $\alpha + \beta + 1 = r$ results in non-redundant number representation systems which cover the conventional radix- r system with $\alpha = 0$, $\beta = r - 1$ as a special case. GSD number systems cover the following special cases.

1. *Binary stored-carry (BSC)*: $r = 2$, $\alpha = 0$, $\beta = 2$.
2. *Radix- r stored-carry (SC)*: $\alpha = 0$, $\beta = r$.
3. *Binary stored-borrow (BSB or BSD)*: $r = 2$, $\alpha = \beta = 1$.
4. *Radix- r stored-borrow (SB)*: $\alpha = 1$, $\beta = r - 1$.
5. *Binary stored-carry-or-borrow (BSCB)*: $r = 2$, $\alpha = 1$, $\beta = 2$.
6. *Radix- r stored-carry-or-borrow (SCB)*: $\alpha = 1$, $\beta = r$.
7. *Minimally redundant symmetric signed-digit*: $2\alpha = 2\beta = r \geq 4$.
8. *Ordinary signed-digit (OSD)*: $r \geq 3$, $\frac{1}{2}r < \alpha = \beta < r$.
 Minimally redundant: $\alpha = \beta = \lfloor \frac{1}{2}r \rfloor + 1$.
 Maximally redundant: $\alpha = \beta = r - 1$.

Radix- r stored-carry number representation systems use the digit set $\{0, 1, 2, \dots, r\}$. The special case $r = 2$ leads to the binary stored-carry (BSC) number system. The main use of BSC numbers is in multioperand addition (multiplication). A BSC number can be added to a conventional binary number, producing a BSC result, using a set of full adders without carry propagation. The stored-carry number systems have been adopted in many implementations [5–8]. Radix- r stored-borrow number systems use the digit set $\{\bar{1}, 0, 1, \dots, r - 1\}$. The special case $r = 2$ leads to the binary stored-borrow (BSB) number system, also known as binary signed-digit (BSD) number system. BSD numbers have been used for representing intermediate temporary values in high-speed multiplication and division algorithms such as Booth's recoding algorithm for multiplication [9]. Two BSD numbers can be added by a limited carry circuit. Implementations using the BSD number system are described in [10–13].

3. The redundant binary approach

In redundant binary logic, each redundant digit is encoded by two or more bits and computation is performed by means of conventional binary logic families such as static or dynamic CMOS. A radix-2 redundant digit, for example, can be encoded by two binary digits as seen in Table 1. The digit set $\{\bar{1}, 0, 1\}$ has two well-known encodings, namely, *sign-mag* and *borrow-save* [14]. In sign-mag the two bits represent a magnitude and a sign, respectively, whereas in borrow-save one bit is positive (*sum*) and the other is negative (*borrow*). This section describes implementations of redundant arithmetic algorithms by means of conventional binary logic circuits.

The main use of redundant binary logic is in multipliers. In high-performance (parallel) multipliers, most of the circuit area is dedicated to adder blocks which sum together partial products to generate the final multiplication result. Consequently, the performance of this type of systems depends heavily on the speed of the adder circuits. Multipliers are very suitable for redundant

Table 1
Radix-2 redundant binary encodings

Encoding	Bit	Values			
Sign-mag	Sign	0	0	1	1
	Mag.	0	1	1	0
	Digit value	0	1	– 1	X
Borrow-save	Borrow	0	1	0	1
	Sum	0	1	1	0
	Digit value	0	0	1	– 1
Carry-save	Carry	0	0	1	1
	Sum	0	1	0	1
	Digit value	0	1	1	2

addition techniques because several stages of addition can be performed without conversion to the standard binary representation. Most high-performance multipliers use a tree structure to increase parallelism in the addition of partial products [15].

3.1. Early tree multipliers

In 1964, Wallace [15] proposed a fully combinational multiplier in which partial products are added by a tree of *pseudoadders*. A pseudoadder sums together three binary numbers and produces two output numbers whose sum equals that of the three inputs. The pseudoadder operates without carry propagation and it can be implemented using full adders, with the third input number being fed to the carry inputs of the full adders and the second output number being formed by the set of carry outputs of the adder cells. By arranging a group of pseudoadders in a tree structure, several additions are performed in parallel. This improves the speed of multiplication and makes the delay proportional to the logarithm of the number of partial products. Fig. 2 depicts a 20-input Wallace tree made of 18 pseudoadders. The pseudoadder is also named a 3:2 *compressor* because it has three inputs and two outputs. A lot of work has been done on the implementation of compressor-based multipliers, and the 4:2 compressor approach is one of the most popular at the present time [16–19].

The Wallace tree approach and other carry-save implementations [6] use redundant binary representation. In these schemes, addition results are expressed as the sum of two numbers. A quantity can therefore be represented in several different ways, thus making the system redundant. If the pair of output numbers is considered as the addition result, it is easy to verify that the pseudoadder implements totally parallel addition by noting that each digit of the result is function only of the input digits of the corresponding compressor.

To obtain the normal binary representation of the result, it is necessary to use a conventional adder to sum the pair of resulting numbers. This step involves carry propagation, which implies a penalty on performance. In multipliers this problem is compensated by accumulating the speed-up

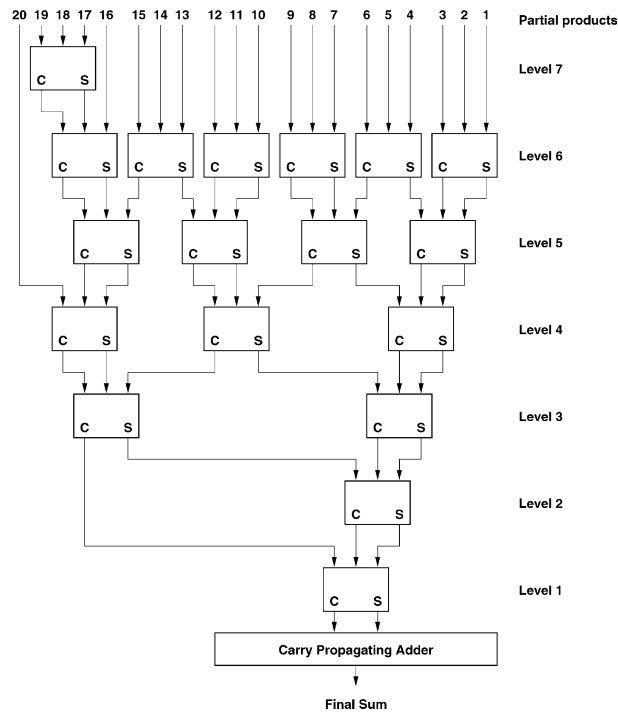


Fig. 2. Wallace tree for a 20-input adder.

of several compression steps before a single conventional addition takes place at the end of the process. The process in which the two resulting numbers are added together can be regarded as a conversion from redundant representation to normal binary representation [11].

In [6], a *carry-save* multiplication algorithm similar to the Wallace tree approach is described. The basic difference between these schemes is that the carry-save adder produces the carry-save sum of two carry-save inputs, while the Wallace tree pseudoadder produces the addition of three binary numbers represented by the sum of two binary numbers. In a unifying way of describing the algorithms one could say that Wallace uses 3:2 compressors and Vuillemin uses 4:2 compressors as the basic addition elements.

Another type of redundant adder tree implementation is based on *borrow-save* or *sign-mag* representations [11]. The main distinguishing characteristic of this approach is that digits can assume negative values. While the digit set in the carry-save scheme is $\{0, 1, 2\}$, the set of digit values in the borrow-save and sign-mag approaches is $\{\bar{1}, 0, 1\}$. This digit set is characteristic of a radix-2 *signed-digit* number system, which was named a *modified signed-digit* number system in [3]. Again, the functional characteristics of this type of algorithms are not very different from those of the carry-save methods. The signed-digit redundant binary adder can also be regarded as a 4:2 compressor.

Compressor-based multioperand addition schemes are very similar to each other. The main difference is in their theoretical basis and the way this drew the path to the corresponding addition algorithm itself. The rest of this section presents practical VLSI implementations of redundant

binary arithmetic schemes. The designs described involve parallel multipliers that implement redundant arithmetic by means of conventional binary logic. There are two main classes of multiplier implementations, taking the theoretical treatment of the arithmetic as the classification criterion, namely, carry-save multipliers and redundant binary multipliers.

3.2. Carry-save multipliers

Most of the implementations under this category are based on Wallace's approach [15]. Here, we describe four of the most representative multiplier designs that have been reported. All of the implementations are based on some type of MOS process (NMOS and CMOS), and all of them use some form of the carry-save adder basic cell.

3.2.1. Pipelined multiplier

The first multiplier circuit to be discussed was published in 1986 by Noll et al. [20]. The multiplier is organized as an array of carry-save adders (3:2 compressors). As seen in Fig. 3, the tree architecture is not followed in this implementation. Avoiding the tree architecture allows for a much more regular layout, which helps reducing the design cycle time and increasing integration. However, using the array architecture comes at the cost of increased latency. This implementation offers a high throughput because the system is *maximally pipelined*, which means that there is a storage element after each adder level in the array. In this way, the clock period has to be sufficiently large to allow the propagation of the signal only through a single compressor cell. This multiplier can operate up to 330 MHz; a very good performance, considering the technology being used. According to the authors, high performance is achieved by making a more efficient use of the hardware. In the maximal pipelining approach, at any given time, all the compressor levels are performing a computation. This contrasts with the non-piped approach, where only one of the compressor levels is active at a time.

Noll's approach was demonstrated by means of an 8×8 -bit multiplier prototype. Fig. 3 depicts a block diagram of the test chip. Note that the circuit only computes the most significant 8 bits of the 16-bit result. One of the contributions of this work is an improved carry-save cell design (shown in Fig. 4). This cell design excludes the time for charging and discharging the multiplier lines (inputs x and y) from the critical path. The test circuit was developed using a $1 \mu\text{m}$ NMOS technology. The circuit consists of 5480 MOSFET transistors, and the active area is 0.6 mm^2 . The power dissipation is 1.5 W, with a supply voltage of 3 V. An operating frequency of 330 MHz at room temperature was achieved, and a latency time of 55 ns is produced by the 18 pipeline stages.

3.2.2. Stanford pipelined iterative multiplier

Another interesting implementation of the carry-save approach is the Stanford pipelined iterative multiplier (SPIM), presented by Santoro et al. in 1989 [16]. In this case, the main goal of the design effort was to develop a multiplier architecture which was faster and more area efficient than a conventional array. Santoro's architecture combines the pipelined Wallace tree approach using 4:2 compressors and an iterative accumulation approach to implement a 64×64 -bit multiplier. By using iterative accumulation, the size of the circuit is significantly reduced while performance specifications are still met. SPIM was able to provide twice the performance of a comparable conventional full array at one-fourth the silicon area [16].

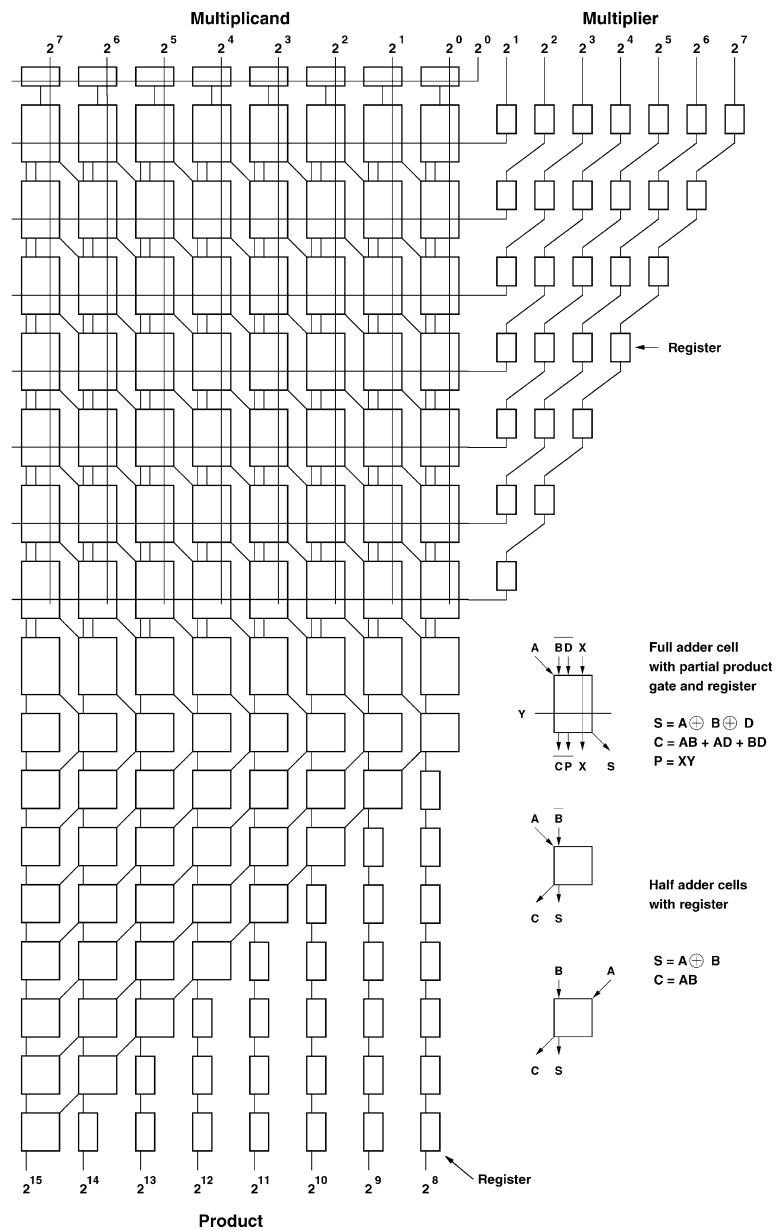


Fig. 3. Block diagram of an 8 × 8-bit multiplier in Noll's implementation.

Fig. 5(a) shows the block diagram of the SPIM datapath. Booth encoders reduce the number of partial products by half [9] – in this implementation the circuit encodes 16 bits per cycle. The Booth-encoded bits control the Booth MUXs in blocks A and B. The A and B MUX outputs drive an eight-input Wallace-tree of 4:2 compressors. Each pipe stage uses one 4:2 compressor. The D block is a carry-save accumulator which also contains a 16-bit hard-wired right shift to align the

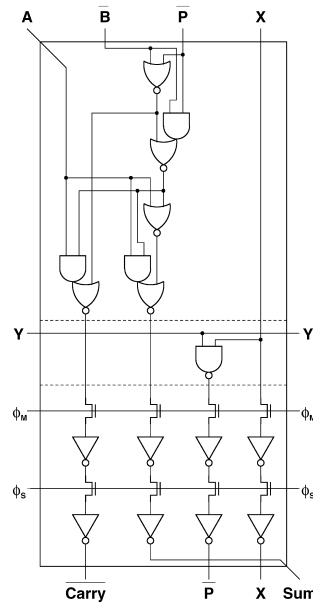


Fig. 4. Novel carry-save cell with pipelined partial product bit.

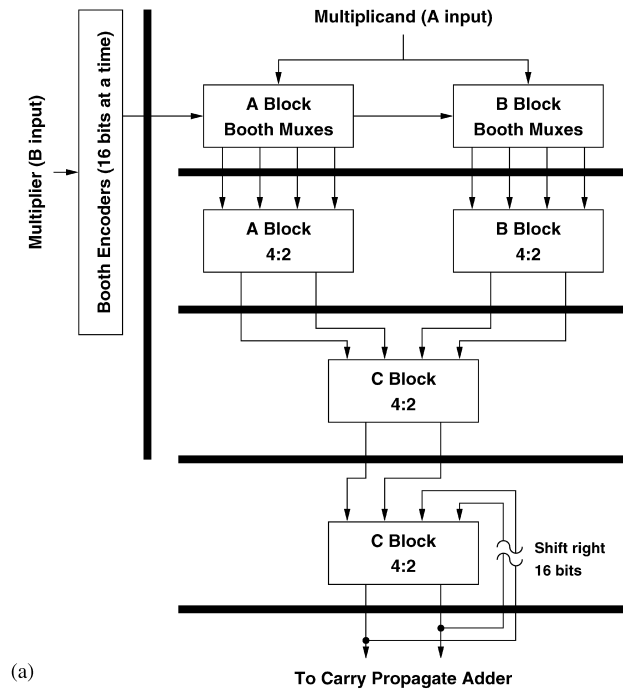
partial sum from the previous cycle to the current partial sum to be added. The pipeline registers are indicated as black bars in the block diagram of Fig 5(a), and the multiplication process can be appreciated in Fig. 5(b).

The latency of the multiplier is 7 cycles, but the circuit can be clocked at high speed due to the short length of each pipe stage. The combined result of the approach is a somewhat better multiplication delay, as compared to other designs, using a very compact circuit. Needing only one-fourth of the area used by its counterparts, this design achieves significant savings in hardware. This factor is very important in VLSI systems like microprocessors, where as many complex systems as possible have to be integrated in a single chip. For Santoro's scheme to work, it is necessary to clock the multiplier at a higher rate than the rest of the system. To solve this problem, the multiplier uses a controllable on-chip clock generator.

SPIM was implemented using a 1.6 μm CMOS process. The core of the chip has 41,000 transistors and a size of $3.8 \times 6.5 \text{ mm}^2$. The on-chip clock generator runs at 85 MHz, and the latency for a 64×64 -bit multiply is under 120 ns with a pipeline rate of one multiply every 47 ns. The latter rate is obtained considering that a multiply operation can be started every four cycles. Fig. 6 shows the design of the basic 4:2 compressor cell. One possible implementation of this cell is based on using two 3:2 carry-save adder cells. This design is used to ease the analysis of performance and comparison with other multiplier designs.

3.2.3. CMOS multiplier with improved parallel structure

An important category of multiplier design is that in which the main objective is to achieve high-performance, without having strong constraints in resources (circuit area) as is the case in SPIM. An important instance of this type of multiplier is the work by Nagamatsu et al. [17,18].

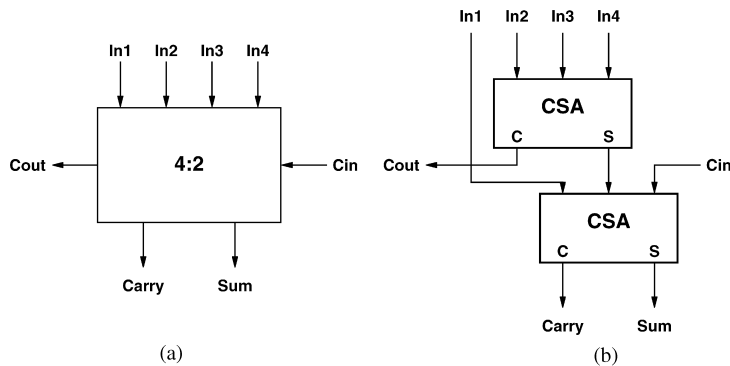


(a)

Action	Cycle							
	0	1	2	3	4	5	6	7
Booth Encode	0–15 startup	16–31	32–47	48–63				
A and B block Booth MUXs		0–15	16–31	32–47	48–63			
A block CSAs			0–7	16–23	32–39	48–55		
B block CSAs			8–15	24–31	40–47	56–63		
C block				0–15	16–31	32–47	48–63	
D block					0–15 clear	16–31	32–47	48–63

(b)

Fig. 5. SPIM datapath. (a) Block diagram. (b) Pipe timing.



(a)

(b)

Fig. 6. The 4:2 compressor basic cell. (a) Block diagram. (b) A 4:2 compressor implemented with two carry-save adders.

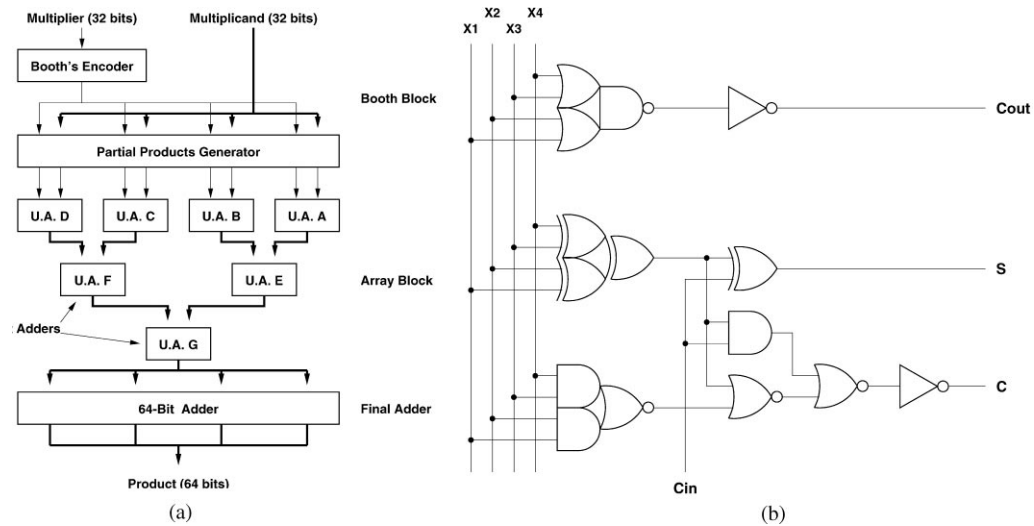


Fig. 7. Tree multiplier architecture implemented by Nagamatsu. (a) Block diagram. (b) Logic circuit diagram of the 4:2 compressor.

The main characteristic of these implementations is that the complete Wallace tree of compressors is built and it is operated in a fully combinational fashion. In this way, the latency of the multiplication operation is reduced to the minimum because a single step is required, and because there is no extra delay introduced by pipeline latches.

Nagamatsu et al. built a 32×32 -bit multiplier [17] applying the modified Booth algorithm to reduce the number of partial products by half, and using a Wallace tree of 4:2 compressors to sum the partial products as seen Fig. 7(a). The 64-bit adder used to obtain the final result relies on the *carry-select* technique which helps reducing carry propagation. Fig. 7(b) shows the logic circuit of the 4:2 compressor. With this approach, the authors were able to obtain a multiply time of 15 ns, the best reported performance up to that time. The test chip has a core of 27,704 transistors in an area of $2.68 \times 2.71 \text{ mm}^2$, and it was fabricated in a $0.8 \text{ }\mu\text{m}$, triple-level interconnect CMOS process. The power dissipation reported by the authors was 277 mW at an operating frequency of 10 MHz (clock for input and output registers).

Mori et al. presented an improved version of their previous work in [18]. In this case, the size of the multiplier was increased to 54×54 bits, but the basic approach remained the same. The organization of the multiplier also consisted of a tree of 4:2 compressors that sum the set of partial products. Booth encoding of the multiplier operand was also adopted. The main architectural difference lies on the design of the 108-bit final adder. Instead of using only the carry-select approach for reducing carry propagation, the new version of the multiplier implemented a combination of the carry-select and the *carry lookahead* (CLA) addition methods. Three main innovations helped the new version of the multiplier achieving better performance. These innovations are (a) the use of a more advanced fabrication process technology with reduced minimum feature size ($0.5 \text{ }\mu\text{m}$), (b) an improved 4:2 basic compressor cell circuit design based on pass transistor logic, and (c) a modified 108-bit final adder design. Fig. 8 depicts the 4:2 compressor circuit. An improved circuit design for the logic gate, shown in Fig. 8, was used to reduce the propagation delay of the 4:2

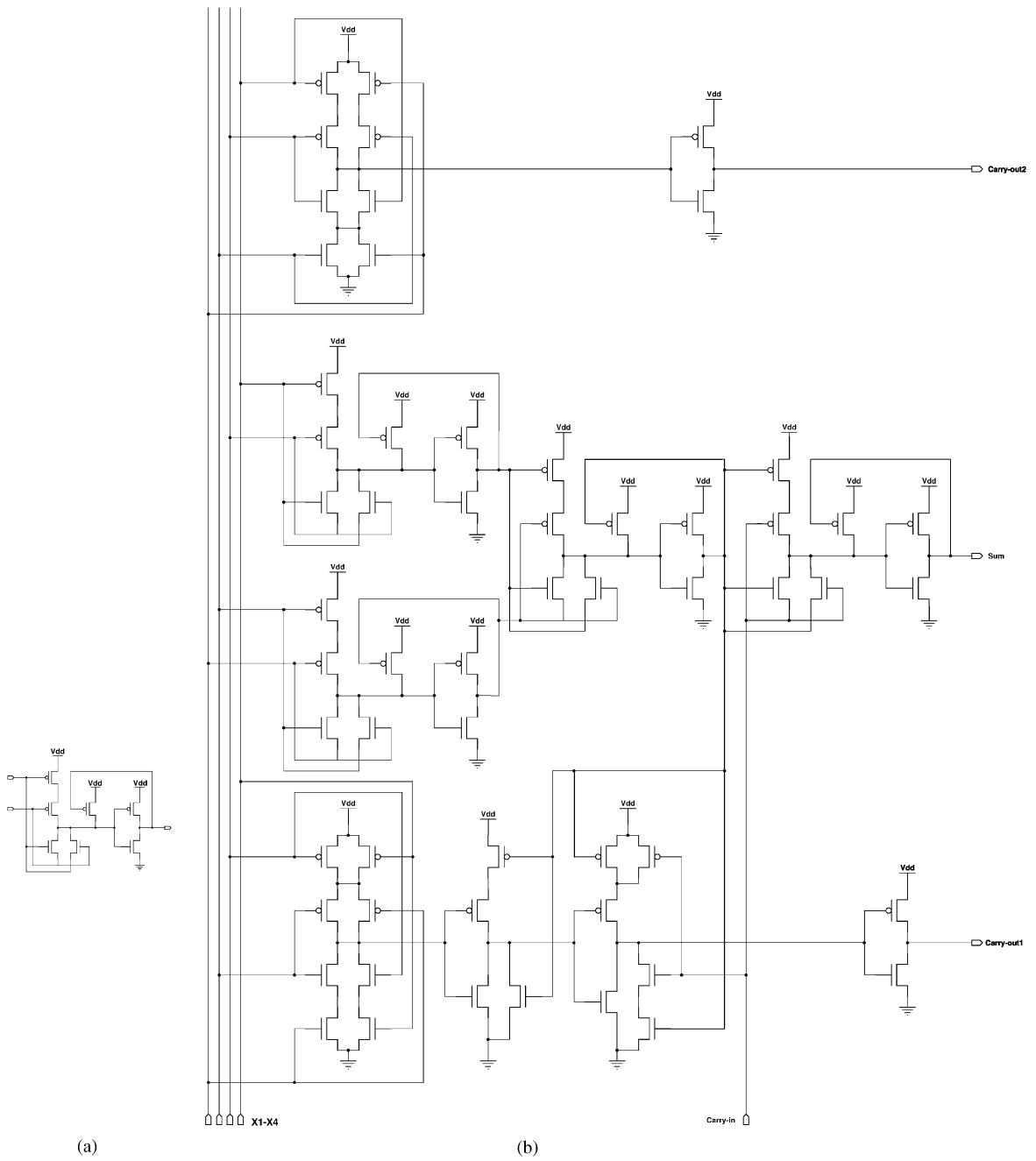


Fig. 8. Circuit design of the basic compressor cell. (a) XOR circuit. (b) 4:2 compressor design.

compressor basic cell. Besides shortening the simulated gate delay to 87.5% of that in the conventional approach, the number of transistors is reduced from 10 to 7. Using the proposed approach, the new multiplier achieved a multiplication delay time of 10 ns. The test chip required 81,600 transistors, and its circuit area was $3.62 \times 3.45 \text{ mm}^2$. The test circuit dissipated 870 mW when clocking the input and output registers at 100 MHz.

3.2.4. Regularly structured tree multiplier

Another multiplier implementation which uses the Wallace tree approach is the 54×54 -bit regularly structured tree multiplier [19], proposed by Goto et al. From the architectural point of view, this multiplier is very similar to those that have been described here [17,18]. Goto's multiplier uses Booth recoding and a Wallace tree of 4:2 compressors. An important distinction of this implementation, from the architectural point of view, is the use of a *Manchester* adder scheme in the final adder (CPA). The main contribution of this work stems from the proposed layout design methodology. To simplify the design process, the authors divide the tree into subcircuit modules that are reused in the construction of the complete tree. The key of the approach is that the wiring scheme is repeated in the modules. In this way, the proposed design method solves one of the most serious problems of Wallace tree multipliers, that is, complicated layout and wiring design. The Wallace tree used to have the drawback of being difficult to layout, due to the irregularity of interconnection among the compressors of the tree. This drawback is eliminated in Goto's implementation, however, by the design of subcircuits that include a wiring scheme that allows them to be replicated in the circuit layout. Two important improvements are obtained by this approach. First, the design cycle is significantly reduced by using tightly coupled, recurring blocks. Second, the resulting layout is better because it features shorter interconnect lengths, reduced circuit area, and higher speed performance. Fig 9 shows the division scheme of the multiplier and depicts the building blocks devised by the authors. In the figure, block 7D is the basic block, and two of them are used to construct block 14D. Similarly, two 14D blocks constitute the complete tree.

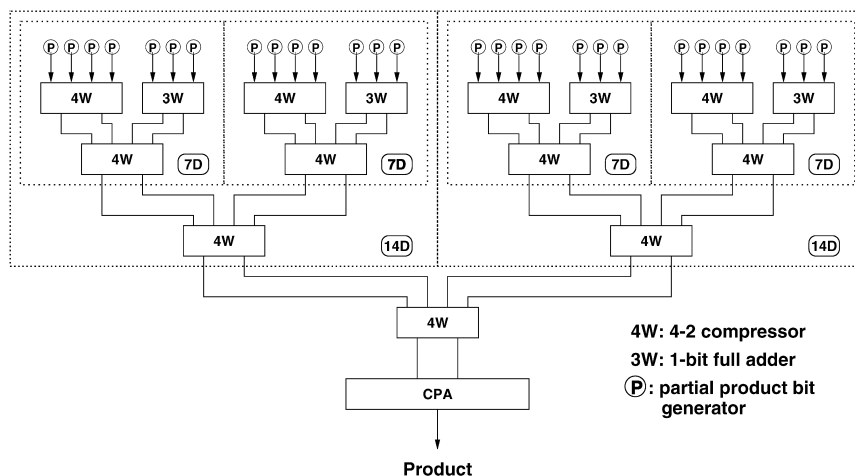


Fig. 9. Block diagram of the 54×54 -bit tree multiplier. Division scheme of partial-product-bit generators and adders.

Table 2
Summary of carry-save multiplier characteristics

Approach	Ref	Size (bits)	Speed	Technology	Power	Transistors
Pipelined array	[20]	8×8	330 MHz	1.0 μm NMOS	1.5 W	5,480 (core)
SPIM	[16]	64×64	47 ns	1.6 μm CMOS	360 mW @ 85 MHz	41,000 (core)
Parallel structure	[17]	32×32	15 ns	0.8 μm CMOS	277 mW @ 10 MHz	27,704 (core)
Parallel structure	[18]	54×54	10 ns	0.5 μm CMOS	870 mW @ 100 MHz	81,600
Regular structure	[19]	54×54	13 ns	0.8 μm CMOS	875 mW @ 40 MHz	82,500

As in the previous examples, Goto's multiplier was demonstrated using a test chip. In this case, fabrication was done using a 0.8 μm , triple metal, CMOS process. The $3.36 \times 3.85 \text{ mm}^2$ circuit included 82,500 transistors. A multiplication delay time of 13 ns was obtained in the experiments. This result compares well to the fastest multiplication delay of a 0.5 μm implementation, which is of 10 ns. Goto's design is expected to perform even faster than 10 ns with an implementation using the more advanced 0.5 μm CMOS process. In the experiments, power dissipation was 875 mW, with a clock frequency of 40 MHz. An important difference in circuit implementation, with respect to the multiplier in [18], is that Goto's circuits [19] are designed using fully static CMOS logic gates, while the other design uses pass transistor gates. Using fully static CMOS allows Goto's design to reduce power consumption.

Table 2 summarizes the most important characteristics of the carry-save multiplier implementations presented.

3.3. Redundant binary architectures

The redundant binary architecture is very similar to a Wallace tree approach. Takagi et al. pointed out these similarities in [11], and referred to Vuillemin's algorithm [6]. The main difference with respect to Vuillemin's work is that Vuillemin uses the carry-save number representation, with digit set $\{0, 1, 2\}$, and Takagi uses the redundant binary representation, with digit set $\{-1, 0, 1\}$. The redundant binary representation allows an easier implementation of two's complement integer multiplication.

3.3.1. High-speed multiplier using a redundant binary adder tree

Harata et al. presented the first integrated circuit implementation of a multiplier using a redundant binary architecture in 1987 [12]. Harata's circuit uses the redundant binary multiplication algorithm described in [11]. The basic idea of the algorithm is to speed-up multiplication by using a tree of redundant binary adders to realize the addition of the partial products as seen in Fig. 10. This approach makes the multiply delay proportional to the logarithm of the operand size, while in array multipliers this proportion is linear. The proposed idea is very similar to the Wallace tree

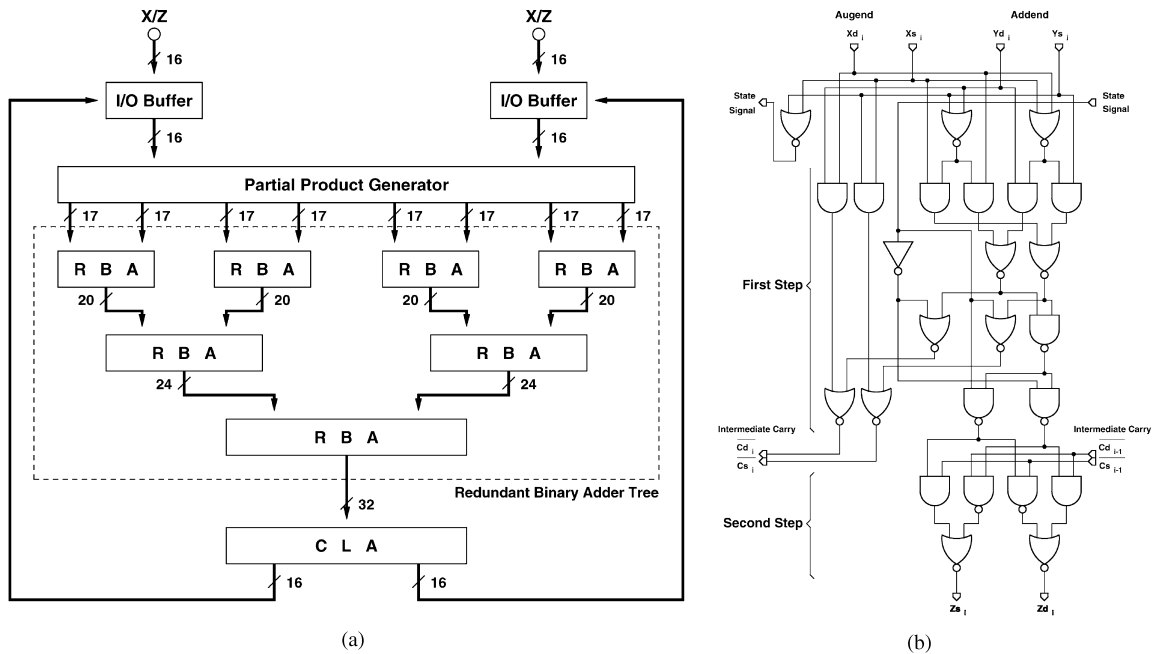


Fig. 10. Diagram of a 16×16 -bit integer multiplier [12]. (a) Block diagram. (b) Redundant binary adder cell.

approach, and it can be said that Harata’s multiplier actually is a Wallace tree. The distinctive characteristic, however, is the realization of the compressor by a redundant binary adder. According to the authors, using a tree of redundant binary adders allows a regular cell array layout implementation, which solves the drawback of layout irregularity in Wallace tree implementations. There is no significant difference between the two approaches in terms of speed performance.

Fig. 10(a) shows the basic architecture of the multiplier clearly depicting the tree structure of the organization. Using this architecture, n partial products are added together in a time proportional to $\log_2 n$. Note that even though the multiplier operands are 16-bits long, the partial product generator block generates only eight, instead of 16, partial products because it uses Booth’s recoding technique [9]. This multiplier internally uses a redundant binary representation as described in Section 2.2. Redundant binary adders (RBA) perform addition of two n -digit redundant binary numbers in a constant delay time, irrespective of n , due to the totally parallel addition characteristic of the number representation. The last block in the organization (CLA) is a carry lookahead adder which converts the resulting sum represented by a redundant binary number into the two’s complement representation of the result. The conversion adder has carry propagation and that is why a fast addition approach such as carry lookahead ($\log_2 n$ delay) is used.

The redundant binary adder cell, seen in Fig. 10(b), determines the performance and the size of the multiplier because it constitutes the delay path of the multiplier and it is the most instantiated cell in the design. The implementation of this adder cell makes Harata’s multiplier a redundant binary design because it uses binary signals to represent signed digits of the set $\{-1, 0, 1\}$. Since the input redundant signals (x_i , y_i , and c_{i-1}) are encoded in binary format, standard binary gates

perform the computations and generate redundant output signals (z_i and c_i) encoded in the assumed binary format. The schematic diagram in Fig. 10(b) shows how the carry output signal is generated without any involvement of the carry input signal – this eliminates the carry signal propagation path usually present in arithmetic circuits.

The approach was demonstrated using a 16×16 -bit multiplier test chip. The authors considered and evaluated three options for realizing the layout of the circuit – they selected the layout topology which allows simple signal flow, makes good use of repeatability, and has good extensibility. The chip was fabricated using a standard enhancement/depletion NMOS process with a $2.7 \mu\text{m}$ design rule. The transistor count of the multiplier is 10,600. The authors report a multiplication time of 120 ns.

3.3.2. High-speed MOS multiplier using redundant binary representation

Kuninobu et al. implemented a high-speed multiplier and divider using redundant binary representation [21]. This multiplier is actually an improved version of the work presented in [11,12]. The approach is basically the same: a multiplier using Booth recoding where the partial products are added together by means of a binary tree of redundant binary adders. This multiplier is part of the floating point unit for a microprocessor built by the authors. This work is interesting because it shows the importance of the multiplication algorithm proposed by Takagi in [11]. There are three main innovations that help this algorithm to keep up to date in the high-performance VLSI world. First, the new circuit implementation uses a $0.8 \mu\text{m}$, 2-layer metal CMOS technology which is far superior to the initial $2.7 \mu\text{m}$ E/D NMOS process technology. Second, the authors use an extended Booth recoding which they call *redundant binary Booth algorithm*. And third, the redundant binary adder cell uses a redesigned logic circuit which is smaller and faster. Fig. 11 presents the logic circuit schematic diagram of the RBA cell in the new multiplier implementation.

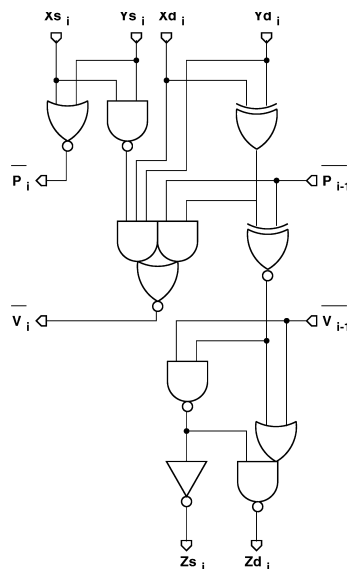


Fig. 11. Logic circuit of the redundant binary adder cell used in [21].

According to the authors, the fabricated microprocessor performs single- and double-precision floating point multiplication in 100 ns.

3.3.3. An 8.8 ns 54×54 -bit multiplier using redundant binary architecture

Makino et al. presented important work on redundant binary arithmetic in [13,22]. Makino's multiplier became the fastest multiplier of the time with an 8.8 ns multiply delay. The implementation is very similar to other redundant binary designs that we have discussed here. The design uses a tree of redundant binary adders to add partial products in a parallel fashion which allows a delay proportional to the logarithm of the operand size. However, the authors identified three problems that affect the performance of redundant binary multipliers. First, additional circuits are used to convert binary numbers to their redundant binary representation. Second, most of the redundant binary adder (RBA) designs are not superior to the 4:2 compressor cells used in non-redundant implementations. And third, the carry propagate adder that converts the redundant binary result into the final binary product is as slow as the conversion adder used in non-redundant approaches. By attacking these problems, the authors obtained a multiplier design of improved performance. The three problems were attacked in the following ways.

Using the *borrow-save* representation, seen in Table 1, helped solving the problem of normal-to-redundant conversion. In borrow-save representation, the digit is represented by the sum of a positive (*sum*) and a negative (*borrow*) bits. In this way, two partial products paired together form a redundant binary number. The conversion process only requires a set of inverters to obtain the complement of the borrow partial product. It is necessary to perform this inversion of bits because the inherent sign of the borrow bits is negative. The bit-wise inversion of the borrow partial product only gives the one's-complement of the number, and the two's-complement is necessary to invert the sign of the number. The authors solve this problem by forming an additional partial product with all the sign bits of the inverted partial products.

Makino proposed and improved design of the redundant binary adder cell in order to attack the problem of inferior redundant binary adder circuits. Fig. 12 depicts the logic diagram of the

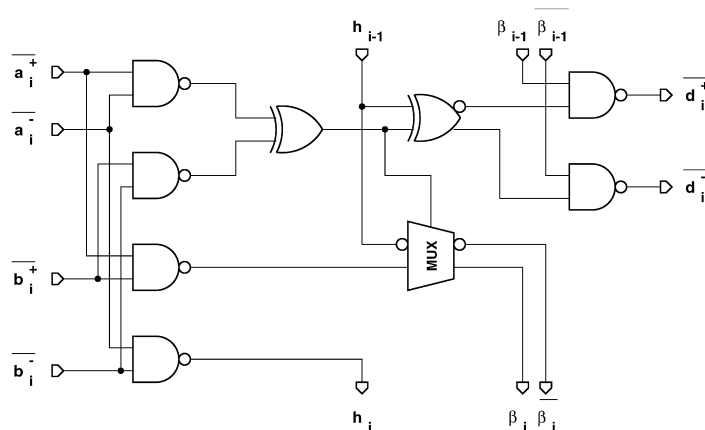


Fig. 12. Schematic diagram of the redundant binary adder cell proposed in [22].

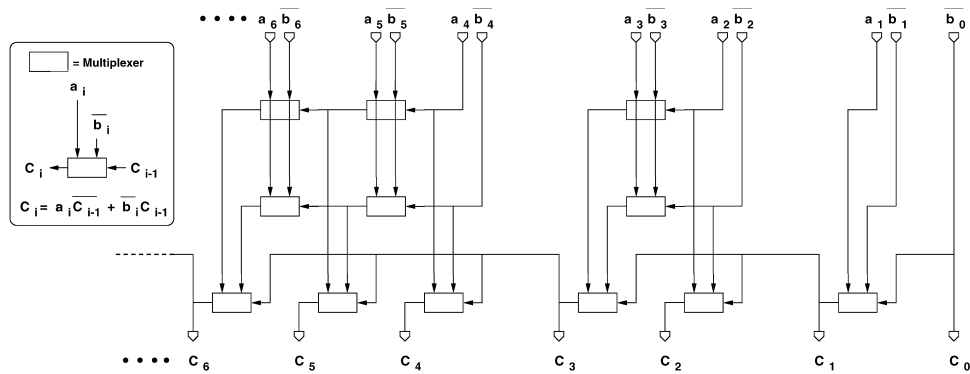


Fig. 13. Block diagram of carry generation circuit in the final output conversion.

Table 3
Summary of redundant binary multiplier characteristics

Approach	Ref	Size (bits)	Speed	Technology	Power	Transistors
RBA Tree	[12]	16×16	120 ns	$2.7 \mu\text{m}$ NMOS	N/A	10,600
RB Representation	[21]	54×60	100 ns ^a	$0.8 \mu\text{m}$ CMOS	4W ^b	1,000,000 ^c
RB Architecture	[22]	54×54	8.8 ns	$0.5 \mu\text{m}$ CMOS	540 mW @ 100 MHz	78,800

^aFor a floating point multiplication.

^bMaximum for the whole processor.

^cIn the whole processor.

proposed redundant binary adder cell. The adder cell design was improved by eliminating OR gates and multi-input complex gates. The authors also took advantage of CMOS circuits by using pass-transistor logic.

Finally the problem of the final representation conversion was solved by designing a conversion method specialized for the redundant-to-normal conversion. The new method uses a carry generation circuit constructed only with simple selector circuits. The design is a kind of *carry-select* method. Fig. 13 shows a block diagram of the carry generation circuit. Besides its reduced delay and transistor count, the new conversion method has the advantage of being easy to layout because the circuit has a regular structure with simple interconnection.

The authors built a 54×54 -bit multiplier chip for demonstration of the multiplication scheme. The circuit uses 78,800 transistors, which is the lowest transistor count reported for this application. The measured multiplication delay was 8.8 ns. Power dissipation was 540 mW with 100 MHz clock speed. The chip was fabricated with a $0.5 \mu\text{m}$, triple metal CMOS process technology.

Table 3 presents the most important characteristics of the redundant binary multiplier circuits described in this section.

4. Current-mode multiple-valued logic designs

In current-mode circuits, signal values are represented by levels of current rather than voltage. Current-mode circuits allow *wired current summation*, where the sum of two or more digits is obtained by the simple connection of wires in a summing node. In this way, the addition of two multivalued current-mode signals is realized without using active elements. Wired current summation is the main motivation for the use of current-mode circuits in arithmetic applications.

4.1. Early current-mode implementations

In 1977, Dao et al. [23] presented one of the first current-mode circuits for multiple-valued logic. In that work, integrated injection logic (I²L) was used for implementing a functionally complete set of multiple-valued functions. Even though no mention is made of redundant arithmetic, the work is a precursor of later current-mode implementations. The main advantages of multiple-valued logic identified in [23] are: high logic density, reduced interconnection area, and high production yields. Dao decided to implement a four-valued logic instead of the more popular three-valued logic due to the better convertibility between four-valued and binary representations. The operating principle is based on threshold logic. Several implementations were proposed, ranging from basic multivalued functions (max, min, complement, successor, and literals) to arithmetic functions (quaternary adder and full product generator) and storage circuits (latches and flip-flops).

Other precursor current-mode implementations are [24,25]. In [24], Freitas et al. presented circuits for conversion of four-valued current-mode signals to and from standard binary signals. In [25], Current et al. described four-valued full adder circuits. Although the circuits proposed do not involve redundant arithmetic, they are very significant because of two reasons. First, the authors identified the advantage of wired summation of logical currents for compact implementation of addition operations [25]. Second, Current and Freitas recognized the importance of making multivalued circuits more practical by using standard fabrication technologies such as CMOS and by using simple operation principles.

Yamakawa developed current-mode circuits for multivalued logic [26] and fuzzy logic [27]. He also recognized the advantage of an easy implementation of addition through current signal summing in current-mode circuits. In [26], an approach for quaternary logic circuits based on MOS devices is described. The approach was named a *hybrid mode* of operation because signals are represented as currents but voltages are used internally in the threshold switches to control pass transistors. Among the various circuits proposed in [26] are a quaternary multiplier and a quaternary divider but these arithmetic circuits do not exploit redundant or signed-digit number systems. In [27], a comprehensive set of fuzzy logic operators was developed using a MOS-based current-mode approach.

The first work on current-mode redundant arithmetic circuits was presented by Kawahito et al. in [28]. Kawahito et al. described the design of a totally parallel adder based on a radix-4, signed-digit, redundant arithmetic system using standard MOS devices. The authors named this circuit a *signed-digit full adder* (SDFFA). An important characteristic of the circuits proposed in [28] is the use of bidirectional currents, that is, currents flowing in two directions. Bidirectional currents are necessary because the signed-digit number system has positive and negative digits, and the

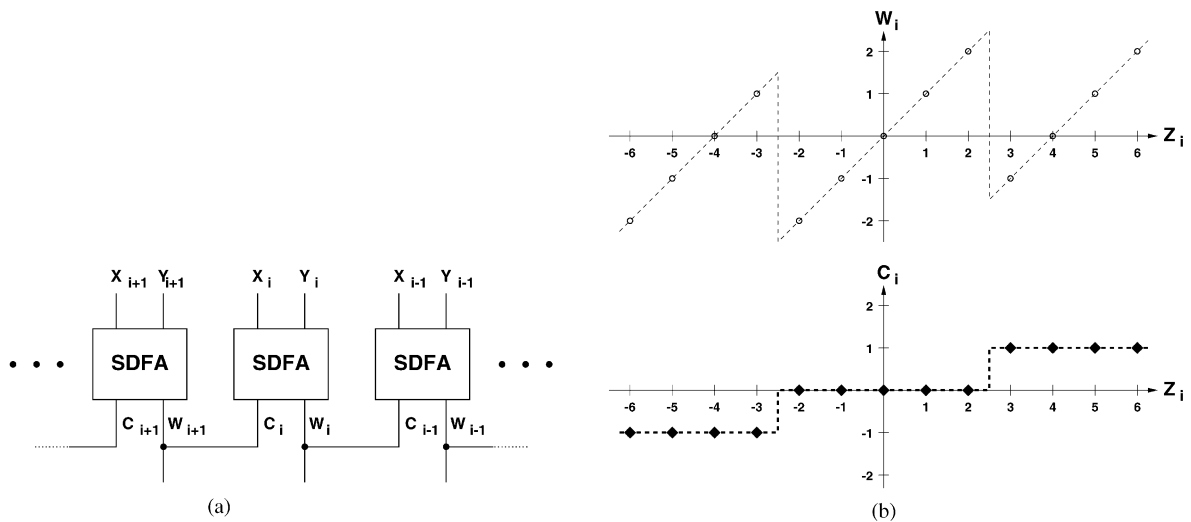


Fig. 14. (a) Block diagram of the signed-digit addition approach developed by Kawahito et al. in [28]. (b) Output functions of the radix-4 signed-digit full adder.

direction of current flow efficiently represents the sign. The work developed by Kawahito et al. is the most comprehensive on current-mode redundant arithmetic circuits [7,8,10,29–39].

4.2. Current-mode redundant arithmetic in Japan

The following paragraphs summarize the work conducted by Shoji Kawahito, Tatsuo Higuchi, and Michitaka Kameyama on current-mode circuit implementations of redundant arithmetic systems.

One of the most important building blocks in redundant arithmetic systems is what Avizienis called a *totally parallel adder* [3]. Kawahito et al. developed a current-mode, radix-4 adder which performs totally parallel addition (signed-digit full adder) [10,28–30,32]. Fig. 14(a) shows a block diagram of the addition approach proposed in [28]. The signed-digit full adder (SDFA) circuit implements the functionality of the generic totally parallel adder described in Section 2.1.1. Specifically, the authors proposed current-mode circuits that realize output functions c_i and w_i according to (1) with $r = 4$. The outputs of the circuit are bidirectional current-mode signals. Therefore, *Step 2* of the addition process seen in Fig. 1 and described by Eq. (2) is achieved by simple wired current summation of signals w_i and c_{i-1} . The output functions of the SDFA cell are depicted in Fig. 14(b), and they are obtained using the expressions given in Section 2.1.2 (with $w_{\min} = -2$ and $w_{\max} = 2$). Note that the input to the SDFA cell is $z_i = x_i + y_i$, and that this signal is also obtained by means of current summation.

Fig. 15(a) shows the block diagram of the current-mode signed-digit adder cell proposed by Kawahito. Block BDI is a *bidirectional current input circuit* whose generic circuit implementation is shown in Fig. 15(b). This block is used to decompose the bidirectional current-mode input signal into a couple of unidirectional current-mode output signals. There is one output (x^+) for positive input flow (I^+) and another output (x^-) for negative input flow (I^-). Block TD is a *threshold*

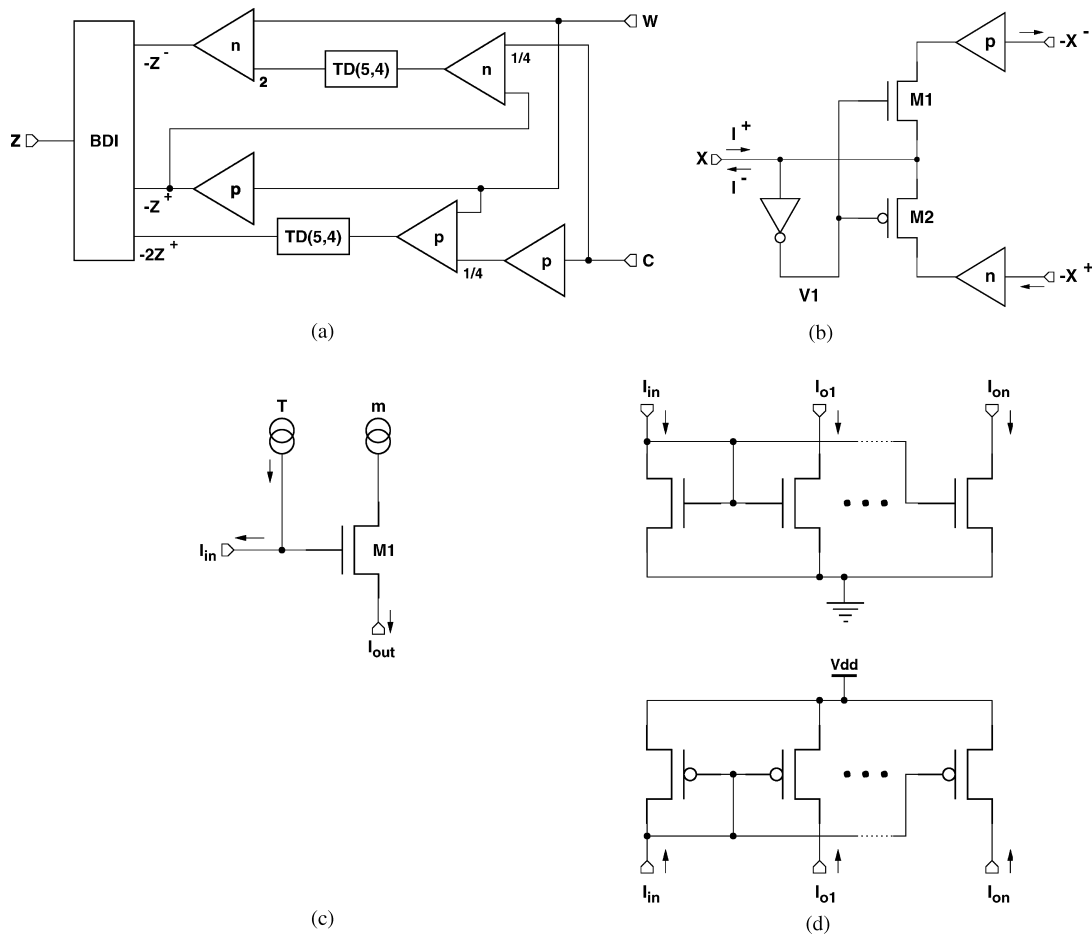


Fig. 15. Current-mode signed-digit full adder design. (a) SDFFA block diagram. (b) BDI circuit. (c) TD circuit. (d) n and p current mirrors.

detector circuit whose circuit is shown in Fig. 15(c). This block provides a current output of m units when its input current is greater than the threshold current, T . Finally, blocks n and p are NMOS and PMOS current mirrors whose circuits are shown in Fig. 15(d). Besides inverting the direction of the input current signal, current mirrors n and p can be used to scale the current level of the input signal or to replicate the input signal so that it can be applied to different nodes. The latter use is required since fanout is restricted to one in current-mode circuits.

The operation of the SDFFA circuit is very simple. The TD blocks in Fig. 15(a) detect the conditions for the generation of the carry signal. One threshold detector works when the input current is positive and the other works when the input current is negative. The interim sum output, w , is obtained by transferring the input current, z , to the output and conditionally adding or subtracting four current units when the output carry is -1 or 1 , respectively. In this way, the transfer function shown in Fig. 14(b) is generated using three regions of operation in the circuit.

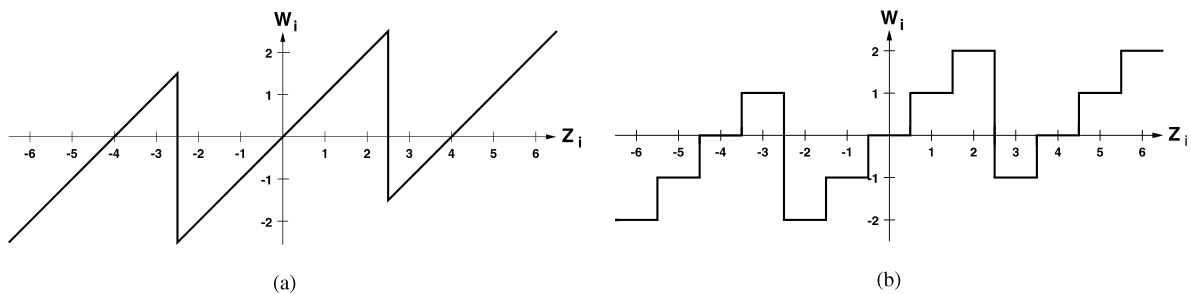


Fig. 16. Signed-digit quantizer. (a) Input and (b) Output.

Other important circuits were necessary to have a complete solution to the signed-digit multivalued implementation problem. The SDFa circuit has very low noise immunity because any variation of input z is reflected at output w . To solve this problem, Kawahito et al. designed a *signed-digit quantizer* (SDQ) [28]. The function of the quantizer is to recover current levels for signals that have been transmitted through several arithmetic modules. With the signed-digit quantizer, output function w of the adder cell is converted from what is shown in Fig. 16(a) to the function depicted in Fig. 16(b). Another important circuit described in [28] is the current-to-binary voltage converter, which obtains the binary sign-magnitude representation of a number in radix-4 current-mode multivalued representation. No separate binary voltage-to-current converter circuit was described in [28]. Instead, since the proposed SDFa cell is intended for building a signed-digit multiplier, the binary-to-current conversion is performed by the product generator circuit.

The circuits developed by Kawahito et al. [28] offer a simple, clean, and efficient approach. The proposed SDFa cell is very compact, but it is not clear that the quantizer can be excluded from the SDFa cell and regarded only as an occasional element in a system. An important problem of the implementation, which is also present in other current-mode circuits, is power consumption. Since the signals are represented as currents, current-mode implementations can consume significant amounts of power. Another problem of the design is in the implementation of the threshold functions. The method for threshold detection used by Kawahito does not provide the gain and noise immunity of voltage-mode binary logic gates and it can be potentially very slow.

A subsequent SDFa design [29] included changes in the transfer functions which allowed for a simplification of the circuit. Using the redundancy of the number system, it was possible to modify the SDFa transfer functions, c and w , at operating points $z = -2$ and 2 . For $z = -2$, the change consisted of making $w = 2$ and $c = -1$ instead of the original $w = -2$, $c = 0$ (see Fig. 14(b)). Similarly, for $z = 2$ the change consisted of making $w = -2$ and $c = 1$ instead of the original $w = 2$, $c = 0$. Note that Eq. (1) still holds for the modified transfer functions. Notice also that, unlike its predecessor, the modified transfer function is an odd function, which allowed the aforementioned circuit simplification. For details on the modified circuit see [29,30]. In addition to refining the initial designs and constructing prototypes to demonstrate their feasibility, the authors also achieved improvements on the device technology aspect. While the first design [28] used a standard CMOS technology, later versions employed a special CMOS process incorporating p-channel depletion MOSFETs [29]. The p-channel depletion MOSFETs are used to implement very compact current sources (see [29]). The authors fabricated a current-mode 32×32 -bit multiplier [32] using the ideas described in the previous sections.

4.2.1. Improvements and evolution of the approach

After presenting the first multivalued current-mode circuits for redundant arithmetic, Kameyama et al. further developed their initial circuits and ideas in order to improve the appeal of the approach. Their efforts were focused at improving characteristics such as speed, noise immunity, power dissipation, and circuit complexity. This section describes two of the approaches aimed at improving multivalued current-mode circuits: *source-coupled logic* and *positive-digit arithmetic*.

Source-coupled logic. Multiple-valued logic current-mode circuits such as those described in the previous section have some disadvantages which become more prominent in deep submicron MOS technologies. First, the delay of a multivalued logic circuit is larger than that of a conventional binary digital circuit because the current in MOS transistors is proportional to the square of the gate voltage. In deep submicron technologies, this problem is aggravated by reduced operating voltages. A second important problem is related to power dissipation, which designers always try to keep as low as possible. Hanyu et al. recognized the necessity of developing new multivalued current-mode circuits with high switching speeds and low operating voltages [37]. Since the largest portion of the delay in current-mode circuits is due to current threshold gates such as the one shown in Fig. 15(c), the authors proposed a new approach named *dual-rail source-coupled logic* which was specifically developed to improve the operation of threshold gate circuits.

Fig. 17(b) shows the new threshold detector presented in [37] while Fig. 17(a) shows the original design. These circuits use depletion p-type MOSFETs for implementing better current sources. The new threshold detector is very similar to the original version of the circuit. The main difference is that current logic signals are represented as differential pairs in the source-coupled threshold detector. Unfortunately, the dual-rail source-coupled design requires routing two wires for each logic signal. Also, in the current threshold detector circuit, it is necessary to have two current comparator transistors (M4 and M5 in Fig. 17(b)) and two output current switches (M7 and M8).

The modified threshold detector circuit has a reduced switching delay due to the current source transistor M6 always being on. In the conventional threshold detector circuit, the largest switching delay is generated in the falling transition of the input current, x . In this transition, the capacitance

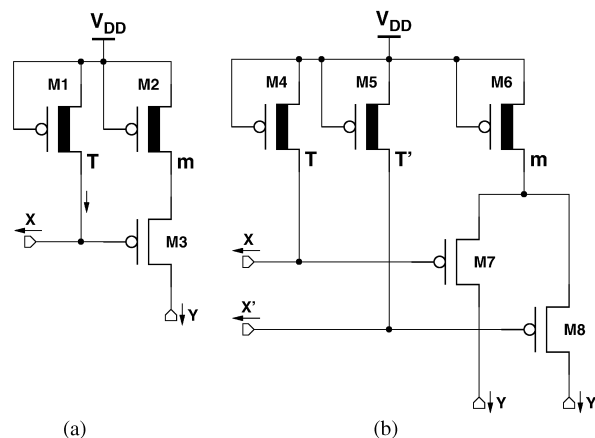


Fig. 17. Current-mode threshold detector circuits. (a) Conventional circuit. (b) Dual-rail source-coupled circuit.

at the gate node of the output switch (M3 in Fig. 17(a)) is charged to a high voltage, which should eventually turn off the PMOS current switch. In this operation, the voltage at the gate of M3 should be large enough to turn off M2 and M3. That is, $V_a = V_{DD} + V_{DS,M2} + V_{GS,M3}$, where $V_{DS,M2} = 0$ because the output current $y = 0$ and $V_{GS,M3} = -V_T$. Here, $-V_T$ represents the threshold voltage of the PMOS transistor M3. Therefore, the capacitance at the gate of M3 should be charged to a voltage level $V_a = V_{DD} - V_T$, and the switching delay is proportional to this voltage level. On the other hand, in the source-coupled threshold detector circuit, the voltage at the gate of the output switch being turned off, M7 or M8, is not required to turn off the current source device, M6, because the operation consists simply of diverting its current to the alternative current switch, M8 or M7. Then, $V_a = V_{DD} + V_{DS,M6} + V_{GS,M7}$, where $V_{DS,M6} = -|V_{TU}|$, V_{TU} is the threshold voltage of the PMOS transistor M6, and $V_{GS,M7} = -V_T$. Therefore, $V_a = V_{DD} - V_T - |V_{TU}|$, which is reduced by $|V_{TU}|$ as compared to the conventional threshold detector design and. Note that the switching delay is directly proportional to V_a .

As it was mentioned, the main advantage of the dual-rail source-coupled approach is on switching delay reduction. It is important to note, however, that this improvement is subject to conditions of matching of the coupled devices and symmetry of the rising and falling input signals, x and x' . Also note that the speed advantage of the new approach comes at the expense of increased circuit and interconnect complexities. The authors evaluated the characteristics of their approach through the implementation of a radix-2 signed-digit adder circuit. This adder was verified by means of simulation, and its performance characteristics were compared to those of the conventional current-mode implementation and binary implementations.

Positive-digit current-mode implementations. A further step in the evolution of current-mode implementations of redundant arithmetic systems consisted of modifying the signed-digit arithmetic proposed in [3]. Kawahito et al. [7] proposed redundant arithmetic circuits based on *positive-digit number representations*. This type of number representation uses digit sets including only positive digits (e.g., $\{0, 1, 2, 3\}$ in radix 2) instead of the symmetric digit sets used by signed-digit systems (e.g., $\{-1, 0, 1\}$ in radix 2). In positive-digit number systems, the redundancy needed for achieving totally parallel addition is obtained by using more digits than the required by conventional radix- r number systems. In other words, the radix- r positive-digit number representations use digit sets of $q + 1$ values of the form $\{0, 1, \dots, r - 1, \dots, q\}$, where $q \geq r$. The main advantage of the positive-digit approach is that it eliminates the use of bidirectional current-mode circuits, thus making the designs more simple. Fig. 18(a) shows a current-mode circuit which implements radix-2, $q = 3$, positive-digit addition. The positive-digit adder cell uses 28 transistors, and the simulation experiments showed a reduction in delay time.

4.3. Multivalued current-mode circuits in France

Even though a significant portion of the research on current-mode redundant arithmetic circuits has been developed in Japan by Kameyama et al., researchers in other parts of the world have made important contributions to the field. We have developed a current-mode, radix-2, signed-digit adder using MOS devices and resonant-tunneling diodes [40]. That work is discussed in Section 5. The following paragraphs describe other current-mode circuits which have been developed in France.

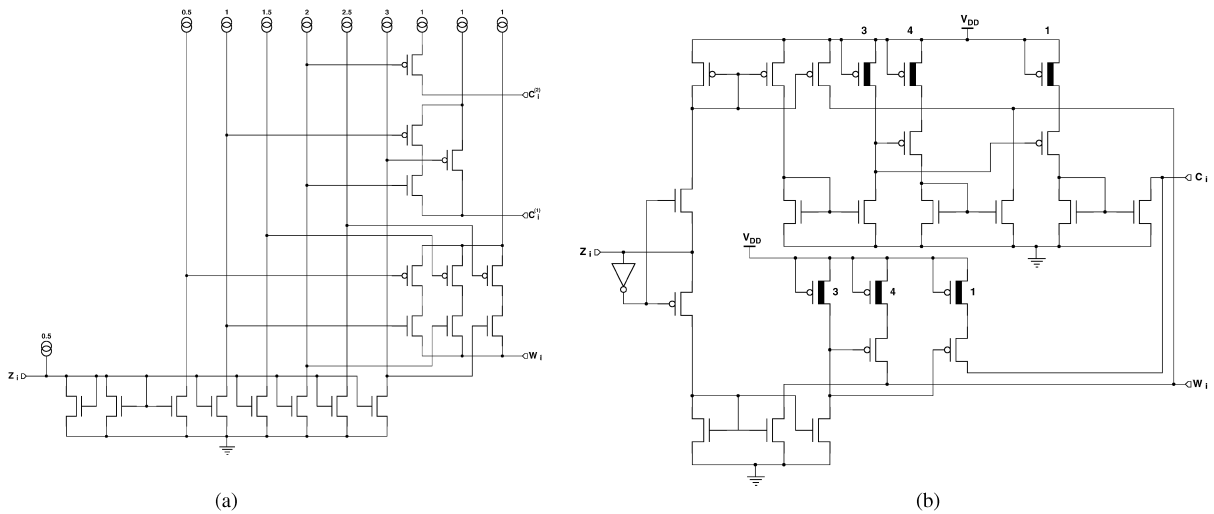


Fig. 18. Schematic circuit diagrams of a (a) radix-2 positive-digit adder cell and (b) its corresponding signed-digit implementation.

Etiemble et al. developed current-mode circuits for redundant arithmetic applications [41,42]. In [41], Etiemble et al. proposed the *limited-carry addition*. This approach is based on the *binary stored-carry* number system. The proposed implementation uses multiple-valued logic and signal representation based on current levels, hence the use of current-mode logic circuits. The binary stored-carry number system is a special case of redundant number representations with radix 2 and the digit set $\{0, 1, 2\}$ (see Section 2.2). The basic approach followed by Etiemble et al. for the implementation of limited-carry adders consists of using a basic functional block, several copies of which can be combined together to form a binary stored-carry adder cell. This approach makes the design task an easy one because only one relatively simple circuit has to be developed and then instantiated several times.

The simplest example of a limited-carry adder is the two-input adder described in [41]. In the binary stored-carry number system, the two inputs of the adder are three-valued current-mode signals. Since the sum p of the two three-valued input signals produces a six-valued result, it is then necessary to first decompose the three-valued operands into their binary components. This task is performed by the *three-valued current input to binary current output converter* (3BC) block. This block is the basic functional unit used to build the two-input binary stored-carry adder. Fig. 19(a) shows the symbol of the 3BC block and the corresponding functional operation. The function table for the 3BC block shows that outputs x^1 and x^0 are, respectively, the carry and sum components of input x . The two-input binary stored-carry adder is built using five 3BC cells, as shown in Fig. 19(b). This design is based on the following expression for the sum of the input operands:

$$p = x + y = 2c_{out}^2 + 2c_{out}^1 + w. \tag{3}$$

Where w is the interim sum and c_{out}^2 and c_{out}^1 are carry output signals. Notice that p is a six-valued signal, and that its binary decomposition works out as follows:

$$p = 2p^1 + p^0 \tag{4}$$

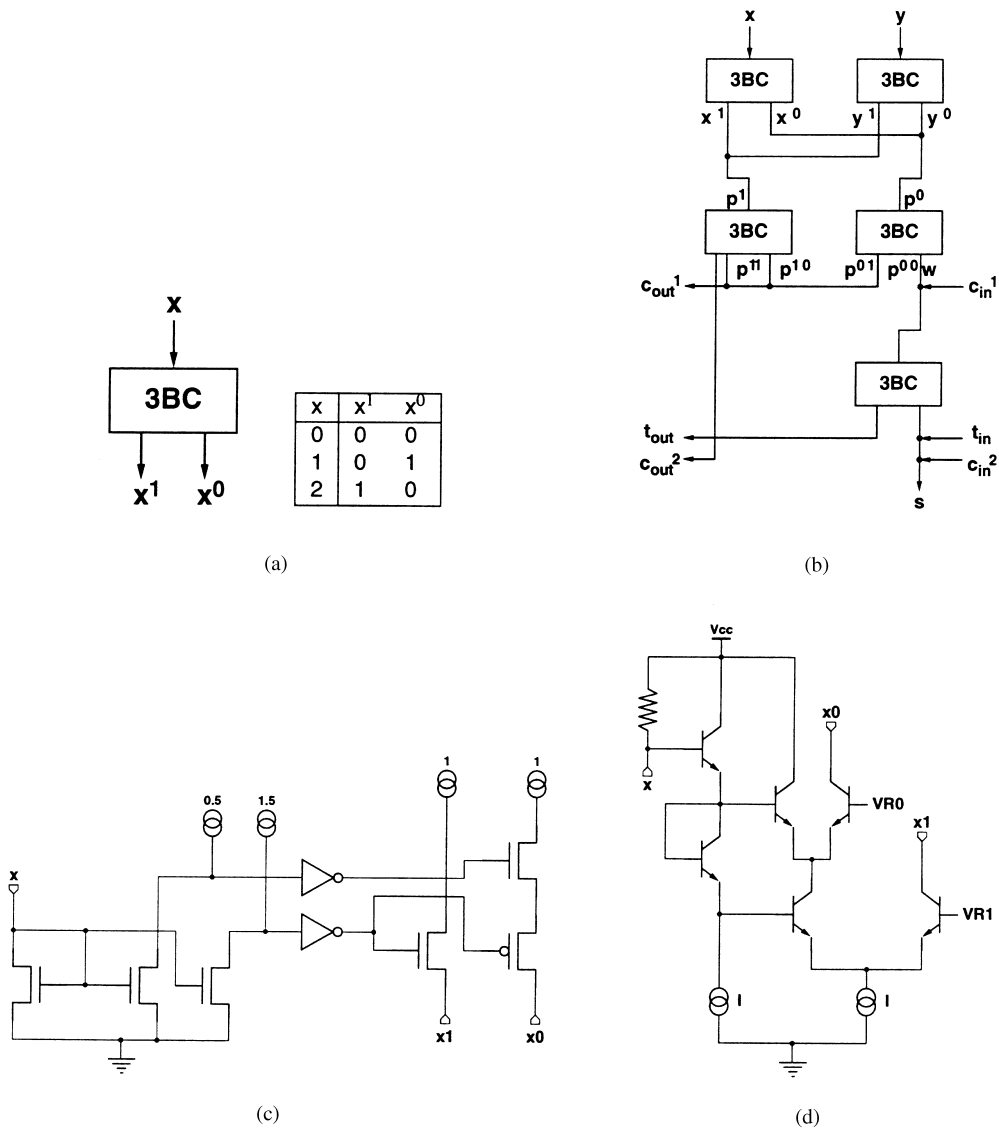


Fig. 19. Current-mode limited-carry adder implementation by Etiemble et al. [41]. (a) 3BC cell, the basic building block of a two-operand adder. (b) Block diagram of the two-input adder. (c) Current-mode CMOS implementation of the 3BC cell. (d) Current-mode ECL 3BC circuit.

$$= 2(2p^1 + p^0)^1 + (2p^1 + p^0)^0 \tag{5}$$

$$= 4p^{11} + 2p^{01} + 2p^{10} + p^{00}. \tag{6}$$

From (3) and (6), the expressions for the carry output signals and the interim sum obtained as

$$c_{out}^2 = p^{11}, \tag{7}$$

$$c_{\text{out}}^1 = p^{11} + p^{10} + p^{01}, \quad (8)$$

$$w = p^{00}. \quad (9)$$

This is the functionality implemented by the top four 3BC cells of the adder shown in Fig. 19(b). The fifth 3BC cell is used to sum w , c_{in}^1 , c_{in}^2 , and t_{in} , and to generate transfer output signal t_{out} .

Circuits implementing the 3BC cell in CMOS and ECL technologies are shown in Figs. 19(a) and 19(b), respectively. Please note that the CMOS 3BC cell circuit is very similar to the current-mode threshold detector in [28]. Actually, two threshold detectors are used in the circuit, one for each of the outputs of the 3BC cell, x^1 and x^0 . The ECL implementation is shown in Fig. 19(d), and it has the disadvantage of requiring voltage references V_{R0} and V_{R1} .

One of the most important contributions of the work presented in [41] is the simplicity and modularity of the design. Circuit complexity has traditionally been one of the drawbacks of multiple-valued logic implementations. The two-input binary stored-carry adder proposed by Etiemble et al. fully exploits the concept of current-mode wired summation. Wired summation of current signals is the only function used in the circuit besides the required multivalued current to binary current conversion performed by the 3BC cells.

Table 4 presents a comparison between current-mode redundant adder implementations. The information contained in the table was obtained from [7,37]. Unfortunately there was no performance information available for the implementations developed by Etiemble et al. [5]. Hanyu reported the construction of layout prototypes for different types of radix-2 signed-digit adders in [37]. Using these prototypes and circuit simulation, Hanyu was able to compare a conventional 34-transistor signed-digit adder with a source-coupled 50-transistor adder built using the same process technology. Rows one and three of Table 4 correspond to the results obtained by Hanyu in his comparison experiment. In [7], the propagation delay of the redundant positive-digit adder is given in terms of equivalent gate delays. Since that work presents a comparison with the delay of the conventional signed-digit current-mode adder, it was possible to estimate the propagation delay of the positive-digit implementation in a 0.8 μm CMOS process using the delay information for the conventional signed-digit adder presented in [37]. The estimated delay value for the positive-digit adder using a 0.8 μm CMOS process is included in the second row of Table 4. From the table it is clear that the positive-digit redundant adder has clear advantages over its counterparts, in terms of propagation delay and device count.

Table 4
Comparison of current-mode redundant adder implementations

Approach	Ref	Delay	Technology	Power	Transistors
Conventional SD	[28]	2.3 ns	0.8 μm CMOS	10.04 mW	34
Positive-digit	[7]	\sim 1.3 ns			28
Source-coupled SD	[37]	1.6 ns	0.8 μm CMOS	10.52 mW	50

5. Other implementations

With MOS technology nearing the limits of device shrinking, the study of alternative fabrication technologies for integrated electronics becomes essential. It is very important to build ultrafast arithmetic circuits. Advanced signal processors with clock rates of the order of gigahertz are required for future system applications such as digital microwave receivers, digital signal processors, and digital RF memories [1]. Carry propagation chains are especially detrimental in ultrafast computation, and they can be completely eliminated by means of redundant arithmetic techniques. However, the sole elimination of carry propagation chains is not sufficient in ultrafast applications like the ones mentioned. It is thus necessary to resort to high-speed integrated circuit technologies and circuit techniques, such as emitter-coupled logic, compound semiconductor devices, and resonant-tunneling quantum electronic devices. This section is divided in two parts. The first part describes emitter-coupled logic implementations of redundant arithmetic. The second part surveys redundant arithmetic circuits based on resonant-tunneling quantum devices and heterostructure devices.

5.1. Implementations using emitter-coupled logic

The emitter-coupled logic (ECL) family was for a long time predominant in high-speed binary logic applications. ECL circuits achieve high speed due to the use of non-saturating transistor operation in differential (emitter-coupled) transistor pairs. Some researchers explored the application of ECL techniques in the implementation of very fast redundant arithmetic circuits [1,43].

In 1987, Luo et al. [43] proposed bipolar ECL circuits for implementing redundant carryless adders using a three-valued logic with the digit set $\{-1, 0, 1\}$. The basic logic building block devised in that work was called the J -operator, which takes one three-valued input and generates one binary output. Depending on the type of J -operator, the output signal is high for one or two of the logic voltage levels at the input of the gate. Consequently, in a three-valued system there can be six different types of J -operators. The voltage levels of the binary output of the J -operator equal the maximum and minimum voltage levels defined for the three-valued logic system. It is easy to observe that described functionality of the J -operator matches that of the multivalued logic literal described by other authors [44]. Luo et al. analyzed the redundant adder functions and found expressions for them in terms of logic combinations of J -operations of the primary inputs $(x_i, y_i, x_{i-1}, y_{i-1})$. A redundant adder circuit was proposed which uses five J -gates and four AND/OR combinational blocks. The authors report the final redundant adder design using 20 ECL gates, including J -operators, AND, and OR gates. Considering that each of these gates requires at least six transistors and a number of resistors, it is easy to find out that the cost of implementation is rather high. The work by Luo et al., however, is a good example of the early search for fast redundant adder implementations using alternative high-speed circuit techniques.

Another application of emitter-coupled circuits in redundant arithmetic was proposed by Lutz J. Micheel in [1]. Searching for very fast arithmetic circuits, Micheel applied ultrafast integrated circuit technologies to multiple-valued ECL circuits in positive-digit and signed-digit arithmetic. Micheel studied the feasibility of pipelined, carry-propagation-free adders and multipliers operating at clock frequencies of the order of two to ten gigahertz using heterojunction bipolar transistors (HBTs) implemented in III/V semiconductor compounds.

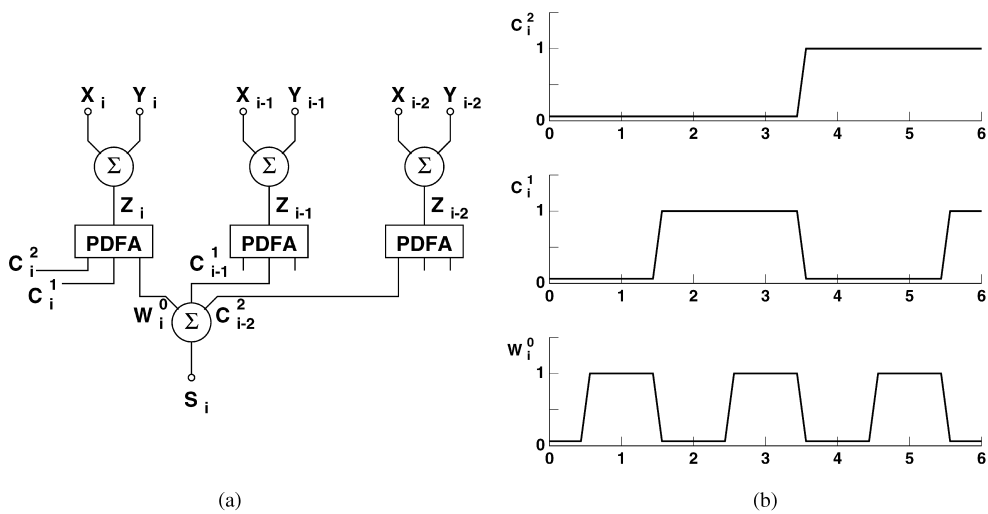


Fig. 20. P2,4 positive-digit adder arithmetic system. (a) Block diagram. (b) PDFAs output functions.

A block diagram of the positive-digit adder circuit proposed in [1] is shown in Fig. 20(a). Micheel's ECL circuit implements a P2,4 arithmetic, which was introduced by Kawahito et al. in [7]. The algorithm P2,4 is a positive-digit arithmetic using the digit set $\{0, 1, 2, 3\}$ with internal operations performed in radix 2. The system shown in Fig. 20(a) accomplishes addition in three steps using two carry transfer digits, C_i^1 and C_i^2 . In order to implement addition correctly, the positive-digit full adder (PDFAs) should have the output functions depicted in Fig. 20(b). As seen in the block diagram, the final summation digit S_i is obtained by adding the interim sum output W_i^0 and one carry transfer digit from each of the two contiguous lower-significance cells, C_{i-1}^1 and C_{i-2}^2 . Micheel proposed the circuit depicted in Fig. 21 to implement the three PDFAs output functions. The circuit is based on the threshold operation of emitter-coupled pairs. In the schematic diagram, the triangular symbol corresponds to a simple emitter-coupled pair comparing the input voltage (left input) to the reference voltage (right input). The larger symbols with two reference inputs represent multivalued logic literal generators made by combining two emitter-coupled pairs. The output pulse (low or high) occurs when the input voltage level lies between the two input threshold voltages. The output pulse is low or high depending on which transistor of the emitter-coupled pair (left or right) the output is taken from. The operating principle of the PDFAs circuit is now easy to explain. A ladder of resistors and transistors generates all the threshold voltages which correspond to switching transitions in the output functions. The threshold emitter-coupled pairs and the literal generators then draw current levels which are combined together by resistors to convert the current summation from the related literals and threshold pairs to voltages.

Based on circuit simulation, Micheel estimated the PDFAs operating clock rates of 1.4–1.6 GHz using an AlGaAs/GaAs HBT system. GaAs HBT devices, however, have a large bandgap of 1.52 eV, a base-emitter turn-on voltage above 1.2 V, and large minimum emitter sizes. As shown in the circuit diagram, it is necessary to connect several base-emitter junctions in series, making high supply voltages necessary and, consequently, increasing power consumption. The authors proposed indium phosphide (InP) technology to solve the problem. With the new technology, the

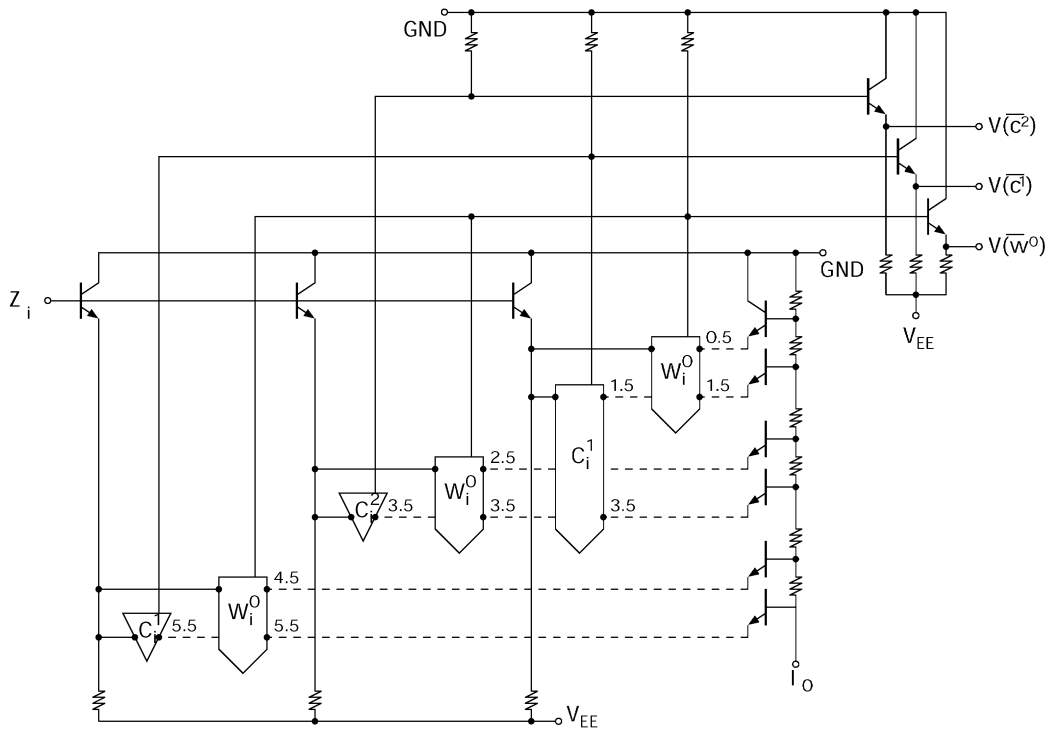


Fig. 21. Positive-digit adder circuit schematic.

base-emitter turn-on voltage can be reduced by 780 millivolts with respect to the GaAs system. The expected operating clock frequencies in the new technology are of the order of 4–10 GHz. Another important disadvantage of the the circuit proposed by Micheel is that it uses a large number of devices.

5.2. Redundant arithmetic using quantum electronic devices

Quantum devices with resonant-tunneling characteristics offer ultrahigh switching speed and dense functionality that can lead to compact, ultrafast circuit implementations [45]. The high functional density of devices such as resonant-tunneling diodes (RTDs) is due to their fold-back I - V characteristics. The effect associated with every folded I - V characteristic is called negative differential-resistance (NDR): a section of the curve where the device current decreases with increasing voltage across the terminals. The presence of one NDR region originates two positive differential-resistance (PDR) regions. Since each PDR region can support one stable circuit state, the RTD is inherently bistable. By stacking several RTDs in an epitaxial process, the devices are connected in series and a multiple-peak characteristic with several PDR regions [46] is obtained. This makes RTDs an important asset in the design of multiple-valued logic circuits where digits assume more than simply two values, as is the case in redundant arithmetic. Many multivalued logic circuits using resonant-tunneling devices have been developed [46–50]. The rest of

this section describes redundant arithmetic implementations using resonant-tunneling quantum devices.

5.2.1. Multivalued redundant arithmetic using nanoelectronic devices

Lutz Micheel et al. described circuit applications of quantum and heterojunction devices in redundant arithmetic logic [51]. Their work concentrates on three-terminal multiple-peak devices such as resonant-tunneling bipolar transistors (RTBTs) and resonant-tunneling field effect transistors (RTFETs). These devices are fabricated by placing resonant-tunneling diodes in the emitter epitaxial stack terminals of heterojunction bipolar or hot electron transistors, respectively. Micheel et al. described a very compact circuit that implements the radix-2 positive-digit algorithm (PD2,4) using multiple-peak RTDs and field effect transistors.

One of the circuits described in [51] implements positive-digit addition of type PD2,4. This algorithm, illustrated in Fig. 20, is identical to the one used in [1]. Fig. 22 shows a simplified schematic diagram of the PD2,4 adder circuit using multiple-peak resonant-tunneling field effect transistors (M-RTFETs). The circuit consists of three identical multilevel folding amplifiers whose inputs are connected by a binary-weighted resistor ladder. Output function W_i is generated by the folding amplifier which is closest to the input. As the input voltage Z_i increases from V_{REF} , the three output voltages start low because V_{REF} turns on the gate voltage of the input FETs in the three folding amplifiers. In the W_i amplifier, when the first valley voltage of the MRTD is reached, the resulting reduction in current flow causes the input FET to switch off. The active load depletion FET then pulls W_i high to V_{DD} . As the input voltage Z_i further increases, this cycle is repeated for all the peak and valley voltages of the MRTD. In this way, the proper interim sum transfer function is generated. Output signals C_{i+1} and C_{i+2} are obtained in a similar fashion. The resistor ladder, however, divides Z_i so that two and four times the input voltage are required to generate the switching transitions of C_{i+1} and C_{i+2} , respectively.

Since the redundant arithmetic circuits proposed by Micheel et al. are at early stages of development, there is no experimental information available on the performance of the proposed circuits yet. The idea proposed by Micheel is to implement these powerful arithmetic circuits using

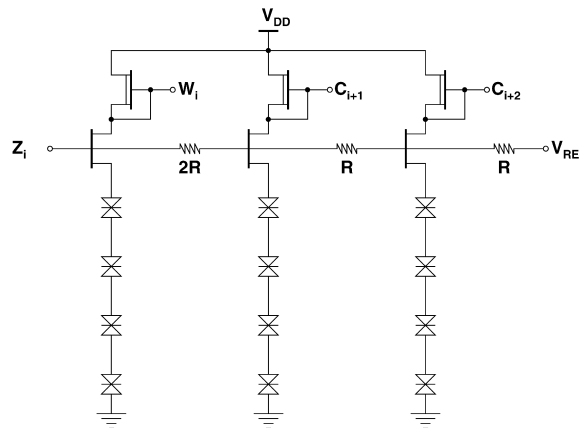


Fig. 22. Positive-digit adder circuit using multiple-peak resonant-tunneling FETs.

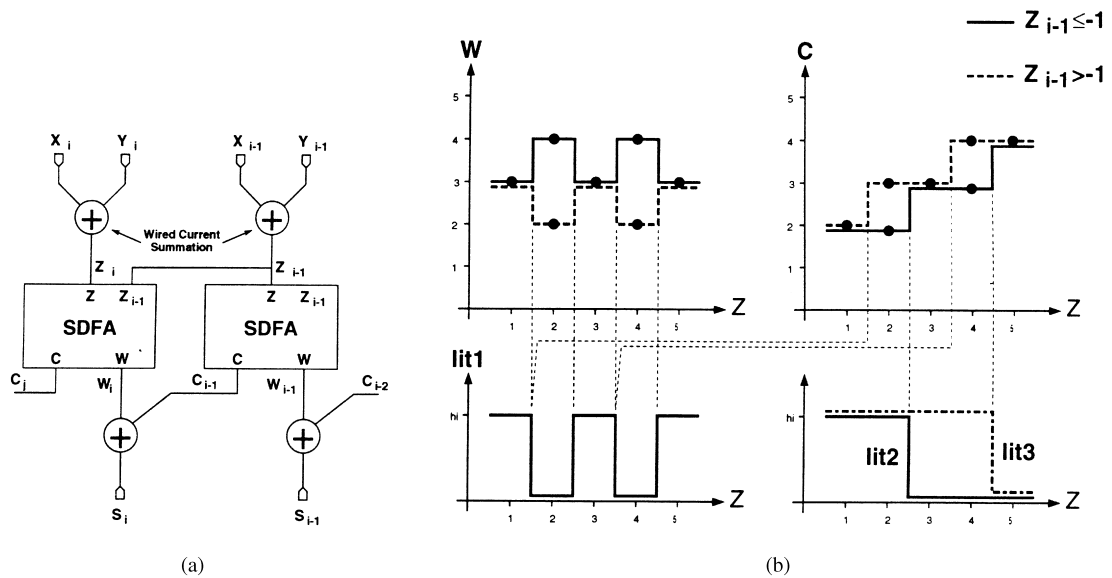


Fig. 23. Totally parallel addition algorithm implemented in [40]. (a) Block diagram. (b) SDFAs output functions.

ultrafast devices with concentrated functionality such as resonant-tunneling diodes and hot electron transistors. This concentrated functionality leads to very efficient circuit implementations requiring few devices. The combination of all these factors is expected to yield very fast, power-efficient redundant arithmetic circuit implementations.

5.2.2. Signed-digit adder using MOS transistors and quantum devices

A new multivalued signed-digit adder uses resonant-tunneling diodes and MOS transistors [40]. A radix-2 arithmetic system was implemented using a three-valued logic with the digit set $\{-1, 0, 1\}$. Even though it is not currently possible to cointegrate RTDs and MOS devices, various efforts towards a technology which will integrate NDR and MOS elements are being conducted [52–54]. It is therefore necessary to develop and study circuits combining the two types of technologies in order to learn how the leading technology can be enhanced by quantum devices.

Fig. 23(a) depicts a block diagram of the signed-digit addition approach used in [40]. Symbols x_i , y_i , c_i , w_i , and s_i represent three-valued, current-mode signals. The addition of x_i and y_i is achieved by wired summation of currents. The function of the SDFAs block is to convert the summation input signal, z , to a two-digit representation of the sum using digits c and w as follows: $rc + w = z$, where $r = 2$. The final sum output, s_i , is obtained by wired current summation of the interim sum output, w_i , and the incoming carry signal, c_{i-1} . Fig. 23(b) shows the transfer functions for the interim sum w and the carry c outputs of the SDFAs cell. All the digits in the diagram are positive because the circuit uses only positive currents. In this case, the digit 0 is represented by a current level “3”, digit -2 is represented by current “1”, and so on. There are two pairs of transfer functions, and the working pair is selected by the value of z_{i-1} . Signal z_{i-1} is used to determine if $c_{i-1} \neq -1$, so that the SDFAs cell is allowed to generate an output $w = -1$ without causing invalid s current levels to be produced by the output wired current summation.

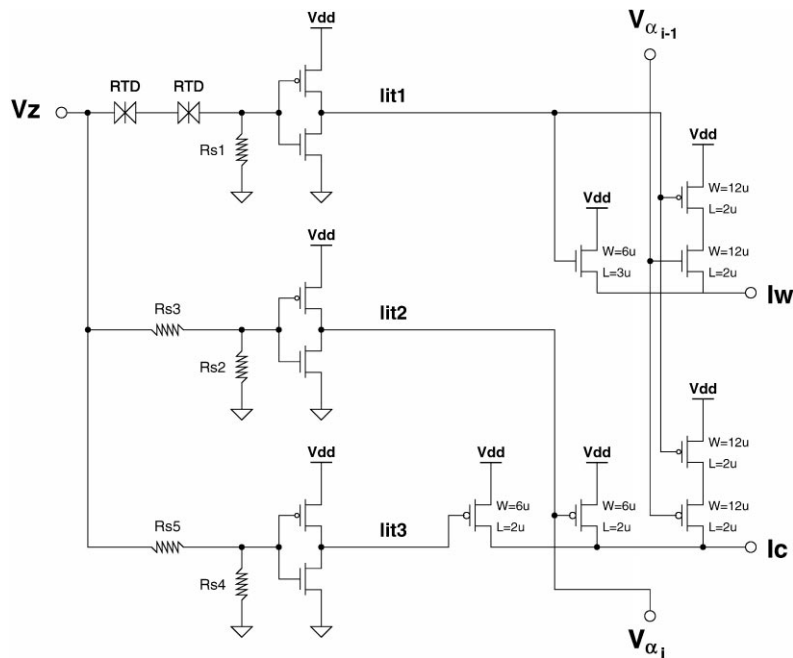


Fig. 24. Circuit diagram of the complete SDFa cell.

As seen in Fig. 23(b), literals *lit1*, *lit2*, and *lit3* contain all the switching threshold points that describe output functions *w* and *c*. The three required literal signals are generated in different blocks of the circuit as seen in Fig. 24. These literal signals are then used to control switched current sources, which in turn synthesize the SDFa output functions. Signal V_α indicates if $z_{i-1} < -1$ and its behavior is identical to literal *lit2*, which also reduces circuit complexity. Fig. 24 shows a circuit diagram of the proposed signed-digit adder. Please note that the circuit includes three literal-generating blocks and two output function synthesizing blocks. There is one current switching block for the interim sum output and one for the carry output. It can be seen in the diagram that literal *lit1* is generated using only a two-peak RTD, a resistor, and a CMOS inverter. This is a very compact implementation considering the sophisticated behavior defined for *lit1* in Fig. 23(b).

The main advantage of the proposed design when compared to other redundant adder implementations is compactness, which is primarily due to the non-linear characteristics of RTDs. Also, current-mode of circuit operation, in which digits are summed by merely connecting their wires together [55], enabled us to reduce the transistor count. Using only 13 CMOS transistors, five resistors, and a two-peak RTD, the total number of active and passive devices used in the proposed SDFa circuit is only 19. From the simulation result, the estimated propagation delay for the circuit is 3.5 ns.

Table 5 presents the most important characteristics of the redundant adder circuits described in this section.

Table 5
Summary of redundant arithmetic circuits using alternative technologies

Approach	Ref	Speed	Technology	Device Count
RB2	[43]	N/A	Silicon Bipolar ECL	150 ^a
PD2,4	[1]	1.6 GHz	GaAs HBT	53 ^b
PD2,4	[51]	N/A	M-RTFETs	21
SDFA	[40]	3.5 ns	RTD + CMOS	19

^aConservative estimate from available information in the paper.

^bEstimated from simplified circuit diagrams.

6. Discussion and conclusions

The preceding sections survey implementations of redundant arithmetic algorithms. For each of the implementations, the operating principle is presented and the main advantages and disadvantages of the approach are discussed. The designs are classified in three main categories, namely, conventional binary logic circuits, current-mode multivalued logic circuits, and circuits based on heterostructure and quantum electronic devices. For each of the identified implementation categories, the designs are evaluated and compared with each other in terms of their speed, power consumption, and the number of devices they require. In this section, we present a general comparison of the implementations which is independent of the design classification.

To compare the implementations of different design categories with each other, it is necessary to adopt a subject of comparison which is common to all the identified design categories. This common element is the single-digit adder cell because it contains similar functional power in all the implementations. In conventional binary logic designs, the single-digit cell corresponds to a 4:2 compressor or a carry-save adder cell (both used in multiplier circuits). In current-mode multivalued logic designs and in heterostructure and quantum electronic implementations, the single-digit cell finds the form of a signed-digit or a positive-digit adder. Table 6 presents the attributes of the different single-digit adder cells surveyed. The table indicates the type of implementation for each entry by separating groups of designs with horizontal lines.

With the possible exception of the algorithm descriptors in the second column, the meaning of the entries in Table 6 should be clear. In the first three rows of the table a specific acronym indicates the type of implementation, where CSA stands for *carry-save adder* and WTC stands for *Wallace-tree compactor*. The other rows use an algorithm descriptor consisting of a two-letter acronym followed by a digit and, optionally, the letter “M”. The two-letter acronym describes the type of algorithm as follows: RB stands for redundant binary, SD stands for signed-digit, and PD stands for positive digit. The digit in the descriptor specifies the radix of the number system being used. Finally, the optional letter “M”, when present, indicates that the implementation uses multiple-valued logic. For example, the algorithm descriptor in the last row of the table specifies a signed-digit, radix-2, multivalued logic implementation.

Some designs were excluded from the comparison in Table 6 because no details of the implementation of the adder cells are included in the original work. Similarly, there were perfor-

Table 6
Comparative summary of redundant adder cells

Algorithm	Reference	Technology	Delay (ns)		Power (mW)	Device count	Year published
CSA	[20]	1.0- μ NMOS	3	* ^a		45	1986
WTC	[17]	0.8- μ CMOS	1.9	*		56	1990
WTC	[18]	0.5- μ CMOS	1.2			58	1991
RB-2	[12]	2.7- μ E/D-NMOS	23	*		74	1987
RB-2	[21]	0.8- μ CMOS	1.3	*		42	1993
RB-2	[22]	0.5- μ CMOS	0.89			56	1996
SD-2-M	[28]	0.8- μ CMOS	2.3		2.51	34	1986
PD-2-M	[7]	0.8- μ CMOS	1.3	*		28	1991
SD-2-M	[37]	0.8- μ CMOS	1.6		2.63	50	1994
RB-2	[43]	Si Bipolar ECL				150	1987
PD-4-M	[1]	GaAs HBT	0.62			53	1992
PD-2-M	[51]	M-RTFETs				21	1993
SD-2-M	[40]	2- μ CMOS + RTD	3.5		2.3	19	1997

^aAsterisks indicate values inferred from published information.

mance characteristics in some of the designs which were not specified, and it was necessary to infer their values from information contained in the paper. Inferred parameter values are marked with an asterisk. It was necessary to leave some blank entries in the table since, in some cases, it was not possible to infer with confidence all the parameters for all the designs and, in other cases [43,51], the designs were only described at the functional and operating principle levels and were not demonstrated experimentally.

Table 6 displays the three types of single-digit cell implementations. The top group in the table corresponds to conventional binary logic designs, the second group includes current-mode MOS implementations, and the bottom group clusters implementations based on heterostructure and quantum devices. Please observe that, as one could expect, there have been many more implementations relying on conventional binary logic than on each of the other two approaches. While six implementations belong to the conventional binary logic group, only three are in the current-mode multivalued logic group. Another interesting observation is that a great majority of the designs surveyed involve MOS-related technologies. In the table, 10 out of 13 designs, 78%, use MOS devices. This is also an expected result given the predominance enjoyed by MOS technology in the world of integrated electronics.

In general, alternative implementations based on current-mode multivalued logic, heterostructure devices, or quantum electronic circuits can be helpful in reducing the number of circuit elements required to build an arithmetic circuit while, at the same time, increasing the speed performance. Moreover, in multivalued logic circuits, the number of interconnecting wires required to achieve certain data bandwidth is reduced. This is of particular importance in VLSI and ULSI

systems where there is an ever-increasing predominance of interconnections on circuit area, speed, and power consumption [47,56]. While alternative techniques offer good prospects for improvement, their development is still at early stages. Being more mature, MOS technology has the advantage of a greater integration capacity and a constant ongoing improvement of its state-of-the-art. It is therefore not possible to conclude from the comparison made in Table 6 that conventional binary logic implementations in MOS technology will be replaced by alternative circuit techniques in the immediate future.

References

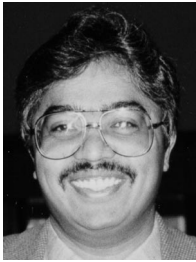
- [1] L.J. Micheel, Heterojunction bipolar technology for emitter-coupled multiple-valued logic in gigahertz adders and multipliers, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1992, pp. 18–26.
- [2] IEEE, IEEE Standard for binary floating-point arithmetic, ANSI/IEEE Standard 754–1985, 1985.
- [3] A. Avizienis, Signed-digit number representations for fast parallel arithmetic, *IRE Trans. Electron. Comput.* (1961) 389–400.
- [4] B. Parhami, Generalized signed-digit number systems: a unifying framework for redundant number representations, *IEEE Trans. Comput.* 39 (1990) 89–98.
- [5] D. Etiemble, K. Navi, Algorithms and multi-valued circuits for the multioperand addition in the binary stored-carry number system, *Proceedings-Symposium on Computer Arithmetic*, IEEE, New York, 1993, pp. 194–201.
- [6] J. Vuillemin, A very fast multiplication algorithm for VLSI implementation, *INTEGRATION, VLSI J.* (1983) 39–52.
- [7] S. Kawahito, K. Mizuno, T. Nakamura, Multiple-valued current-mode arithmetic circuits based on redundant positive-digit number representations, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1991, pp. 330–339.
- [8] S. Kawahito, Y. Mitsui, M. Ishida, T. Nakamura, Parallel hardware algorithms with redundant number representations for multiple-valued arithmetic VLSI, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1992, pp. 337–345.
- [9] A.D. Booth, A signed binary multiplication technique, *Quart. J. Mech. Appl. Math.* 4 (1951) 236–240.
- [10] S. Kawahito, M. Kameyama, T. Higuchi, Multiple-valued radix-2 signed-digit arithmetic circuits for high-performance VLSI systems, *IEEE J. Solid State Circuits* 25 (1990) 125–131.
- [11] N. Takagi, H. Yasuura, S. Yajima, High-speed VLSI multiplication algorithm with a redundant binary addition tree, *IEEE Trans. Comput.* C-34 (1985) 789–796.
- [12] Y. Harata, Y. Nakamura, H. Nagase, M. Takigawa, N. Takagi, A high-speed multiplier using a redundant binary adder tree, *IEEE J. Solid State Circuits* SC-22 (1987) 28–33.
- [13] H. Makino, Y. Nakase, H. Shinohara, A 8.8-ns 54×54 -bit multiplier using new redundant binary architecture, *Proceedings-International Conference on Computer Design: VLSI in Computers and Processors*, IEEE, New York, 1993, pp. 202–205.
- [14] C.N. Lyu, D.W. Matula, Redundant binary booth recoding, *Proceedings-Symposium on Computer Arithmetic*, IEEE, New York, 1995, pp. 50–57.
- [15] C.S. Wallace, A suggestion for a fast multiplier, *IEEE Trans. Electron. Comput.* EC-13 (1964) 14–17.
- [16] M.R. Santoro, M.A. Horowitz, SPIM: a pipelined 64×64 -bit iterative multiplier, *IEEE J. Solid State Circuits* 24 (1989) 487–493.
- [17] M. Nagamatsu, S. Tanaka, J. Mori, K. Hirano, T. Noguchi, K. Hatanaka, A 15-ns 32×32 -b CMOS multiplier with an improved parallel structure, *IEEE J. Solid State Circuits* 25 (1990) 494–497.
- [18] J. Mori, M. Nagamatsu, M. Hirano, S. Tanaka, M. Noda, Y. Toyoshima, K. Hashimoto, H. Hayashida, K. Maeguchi, A 10-ns 54×54 -b parallel structured full array multiplier with 0.5- μm CMOS technology, *IEEE J. Solid State Circuits* 26 (1991) 600–605.

- [19] G. Goto, T. Sato, M. Nakajima, T. Sukemura, A 54×54 -b regularly structured tree multiplier, *IEEE J. Solid State Circuits* 27 (1992) 1229–1236.
- [20] T.G. Noll, D. Schmitt-Landsiedel, H. Klar, G. Enders, A pipelined 330-mhz multiplier, *IEEE J. Solid State Circuits* SC-21 (1996) 411–416.
- [21] S. Kuninobu, T. Nishiyama, T. Taniguchi, High speed MOS multiplier and divider using redundant binary representation and their implementation in a microprocessor, *IEICE Trans. Electron.* E76-C (1993) 436–444.
- [22] H. Makino, Y. Nakase, H. Suzuki, H. Morinaka, H. Shinohara, K. Mashiko, An 8.8-ns 54×54 -bit multiplier with high speed redundant binary architecture, *IEEE J. Solid State Circuits* 31 (1996) 773–783.
- [23] T.T. Dao, E.J. McCluskey, L.K. Russell, Multivalued integrated injection logic, *IEEE Trans. Comput.* C-26 (1977) 1233–1241.
- [24] D.A. Freitas, K.W. Current, A quaternary logic encoder-decoder circuit design using CMOS, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1983, pp. 190–195.
- [25] K.W. Current, D.A. Freitas, F.A. Edwards, CMOS Quaternary threshold logic full adder circuits, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1985, pp. 318–322.
- [26] T. Yamakawa, CMOS multivalued circuits in hybrid mode, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1985, pp. 144–151.
- [27] T. Yamakawa, T. Miki, F. Ueno, The design and fabrication of the current mode fuzzy logic semi-custom IC in the standard CMOS IC technology, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1985, pp. 76–82.
- [28] S. Kawahito, M. Kameyama, T. Higuchi, VLSI-oriented bi-directional current-mode arithmetic circuits based on the radix-4 signed-digit number system, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1986, pp. 70–77.
- [29] S. Kawahito, M. Kameyama, T. Higuchi, H. Yamada, A high-speed compact multiplier based on multiple-valued bi-directional current-mode circuits, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1987, pp. 172–180.
- [30] M. Kameyama, S. Kawahito, T. Higuchi, A multiplier chip with multiple-valued bidirectional current-mode logic circuits, *Computer* 21 (1988) 43–56.
- [31] M. Kameyama, T. Sekibe, T. Higuchi, Design of highly parallel residue arithmetic circuits based on multiple-valued bidirectional current-mode MOS technology, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1988, pp. 6–13.
- [32] S. Kawahito, M. Kameyama, T. Higuchi, H. Yamada, A 32×32 -bit multiplier using multiple-valued MOS current-mode circuits, *IEEE J. Solid State Circuits* 23 (1988) 124–132.
- [33] M. Kameyama, T. Sekibe, T. Higuchi, Highly parallel residue arithmetic chip based on multiple-valued bidirectional current-mode logic, *IEEE J. Solid State Circuits* 24 (1989) 1404–1411.
- [34] M. Kameyama, M. Nomura, T. Higuchi, Modular design of multiple-valued arithmetic VLSI system using signed-digit number system, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1990, pp. 355–362.
- [35] M. Honda, M. Kameyama, T. Higuchi, Residue arithmetic based multiple-valued VLSI image processor, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1992, pp. 330–336.
- [36] S. Kawahito, Y. Mitsui, T. Nakamura, VLSI-oriented multiple-valued current-mode arithmetic circuits using redundant number representations, *IEICE Trans. Electron.* E76-C (1993) 446–454.
- [37] T. Hanyu, A. Mochizuki, M. Kameyama, Multiple-valued current-mode MOS integrated circuits based on dual-rail source-coupled logic, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1994, pp. 19–26.
- [38] T. Hanyu, A. Mochizuki, M. Kameyama, A 1.5-v supply 200 MHz pipelined multiplier using multiple-valued current-mode MOS differential logic circuits, *Proceedings of the International Solid-State Circuits Conference*, IEEE, New York, 1995, pp. 314–315.
- [39] T. Hanyu, A. Mochizuki, M. Kameyama, Multiple-valued arithmetic integrated circuits based on 1.5V-supply dual-rail source-coupled logic, *Proceedings of the International Symposium on Multiple-Valued Logic*, IEEE, New York, 1995, pp. 64–69.

- [40] A.F. González, P. Mazumder, Compact signed-digit adder using multiple-valued logic, Proceedings of the 17th Conference on Advanced Research on VLSI, IEEE, New York, 1997, pp. 96–113.
- [41] D. Etiemble, K. Navi, A basis for the comparison of binary and m-valued current mode circuits: the multioperand addition with redundant number systems, Proceedings of the International Symposium on Multiple-Valued Logic, IEEE, New York, 1993, pp. 216–221.
- [42] A. Kazeminejad, K. Navi, D. Etiemble, CML current mode full adders for 2.5-v power supply, Proceedings of the International Symposium on Multiple-Valued Logic, IEEE, New York, 1994, pp. 10–14.
- [43] Y.-F. Luo, T.-Y. Chen, Totally parallel algorithm for symmetric redundant binary number system and its implementation, Proceedings of the International Symposium on Multiple-Valued Logic, IEEE, New York, 1987, pp. 318–324.
- [44] A.K. Jain, R.J. Bolton, M.J. Abd-El-Barr, CMOS multiple-valued logic design-Part I: Circuit implementation, IEEE Trans. Circuits Systems-I: Fund. Theory Appl. 40 (1993) 503–514.
- [45] F. Capasso, S. Sen, F. Beltram, L.M. Lunardi, A.S. Vengurlekar, P.R. Smith, N.J. Shah, R.J. Malik, A.Y. Cho, Quantum functional devices: Resonant-tunneling transistors, circuits with reduced complexity, and multiple-valued logic, IEEE Trans. Electron Devices 36 (1989) 2065–2082.
- [46] A.C. Seabaugh, Y.-C. Kao, H.-T. Yuan, Nine-state resonant tunneling diode memory, IEEE J. Solid State Circuits 13 (1992) 479–481.
- [47] T. Hanyu, Y. Yabe, M. Kameyama, Multiple-valued programmable logic array based on a resonant tunneling diode model, IEICE Trans. Electron. E76-C (1993) 1126–1132.
- [48] H.C. Lin, Resonant tunneling diodes for multi-valued digital applications, Proceedings of the International Symposium on Multiple-Valued Logic, IEEE, New York, 1994, pp. 188–195.
- [49] T. Waho, Resonant tunneling transistor and its application to multiple-valued logic circuits, Proceedings of the International Symposium on Multiple-Valued Logic, IEEE, New York, 1995, pp. 130–138.
- [50] L.J. Micheel, H.L. Hartnagel, Interband RTDs with nanoelectronic HBT-LED structures for multiple-valued logic computation, Proceedings of the International Symposium on Multiple-Valued Logic, IEEE, New York, 1996, pp. 80–85.
- [51] L.J. Micheel, A.H. Taddiken, A.C. Seabaugh, Multiple-valued logic computation circuits using micro- and nanoelectronic devices, Proceedings of the International Symposium on Multiple-Valued Logic, IEEE, New York, 1993, pp. 164–169.
- [52] U. König, M. Kuisl, J.-F. Luy, F. Schäffler, Si/SiGe resonant tunneling devices separated by surrounding polysilicon, Electron. Lett. 25 (1989) 1169–1171.
- [53] U. König, M. Kuisl, F. Schäffler, G. Fischer, T. Kiss, Operating CMOS after a Si-MBE process: a precondition for future three-dimensional circuits, IEEE Electron Device Lett. 11 (1990) 218–220.
- [54] N. Evers, O. Vendier, C. Chun, M.R. Murti, J. Laskar, N.M. Jokerst, T.S. Moise, Y.-C. Kao, Thin film pseudomorphic AlAs/InGaAs/InAs resonant tunneling diodes integrated onto Si substrates, IEEE Electron Device Lett. 17 (1996) 443–445.
- [55] M. Kameyama, T. Higuchi, Design of a radix-4 signed-digit arithmetic circuit for digital filtering, Proceedings of the International Symposium on Multiple-Valued Logic, IEEE, New York, 1980, pp. 272–277.
- [56] M. Shoji, High-speed Digital Circuits, Addison-Wesley, Reading, MA, 1996, pp. 73–135 (Chapter 2).



Alejandro F. González received the B.E. degree in electrical engineering (Licenciado en Ingeniería Electrónica) from the Instituto Tecnológico y de Estudios Superiores de Occidente, Guadalajara, Mexico, in 1993 and the M.S.E. degree in electrical engineering from the University of Michigan, Ann Arbor, in 1995. He is currently pursuing the Ph.D. degree in electrical engineering at the University of Michigan. His research interests include ultrafast digital circuit design, multiple-valued logic, and VLSI design automation.



Pinaki Mazumder received the B.S.E.E. degree from the Indian Institute of Science in 1976, the M.Sc. degree in computer science from the University of Alberta, Canada, in 1985, and the Ph.D. degree in electrical and computer engineering from the University of Illinois at Urbana-Champaign in 1987.

Presently, he is with the Department of Electrical Engineering and Computer Science of the University of Michigan, Ann Arbor as an Associate Professor. Prior to this he was a Research Assistant with the Coordinated Science Laboratory, University of Illinois at Urbana-Champaign for two years and was with Bharat Electronics Ltd. (a collaborator of RCA), India, for over six years, where he developed several types of analog and digital integrated circuits for consumer electronics products. During the summer of 1985 and 1986, he was a Member of Technical Staff in the Indian Hill Branch of AT&T Bell Laboratories. During 1996–1997, he spent his sabbatical leave as a visiting faculty at Stanford University, University of California at Berkeley, and Nippon Telephone and Telegraph, Japan.

His research interests include VLSI testing, physical design automation, and ultrafast circuit design. He has published over 160 papers on these topics in archival journals and proceedings of the international conferences

Dr. Mazumder was a recipient of Digital's Incentives for Excellence Award, BF Goodrich National Collegiate Invention Award, National Science Foundation Research Initiation Award and Bell Northern Research Laboratory Faculty Award. Recently, he has become an IEEE Fellow for his contributions to the field of VLSI.

Dr. Mazumder has lead his research group's efforts in VLSI testing and built-in self-repair techniques and has developed silicon compilers for RAM, ROM and PLA with built-in self-repairable capabilities. Two international patents on this research work are pending now. He is the co-author of two books — "Testing and Testable Design of High-Density Random-Access Memories" (over 450 pages) and "Semiconductor Random-Access Memories: Testing and Reliability" (over 250 pages). He was a Guest Editor of the IEEE Design and Test Magazine's special issue on multimegabit memory testing, March 1993 and the Journal of Electronic Testing – Theory and Applications special issue on memory testing and reliability, June 1994.

Dr. Mazumder has also done quite extensive work in the area of VLSI physical design. He has developed a suite of distributed place-and-route tools for VLSI and FPGA chips. He has co-authored a book with Dr. Elizabeth Rudnick of the University of Illinois, entitled: "Genetic Algorithms for VLSI Design, Layout, and Test Automation," Prentice Hall, 1998, (350 pages).

Dr. Mazumder has worked over six years as an integrated circuit designer in semiconductor companies. He is currently leading ultrafast circuit design activities for nano and quantum electronic devices. He has successfully developed CAD tools for high-performance VLSI circuit simulation (NDR-SPICE) and numerous circuit topologies for quantum MOS and other quantum-well devices. Several US and Japanese semiconductor companies including Texas Instruments, Hughes Research Laboratory, Lockheed Martin, NTT and NEC have been collaborating with him on this research work. He is the guest editor of two special issues on emerging nanoelectronic technologies and their applications in IEEE Transactions on VLSI System (December 1997) and the Proceedings of the IEEE (1998). He is on the Editorial Board of Proceedings of the IEEE, and is also an associate editor of the IEEE Transactions of VLSI.

He is an IEEE Fellow, and member of Sigma Xi, Phi Kappa Phi, and ACM SIGDA.