Signed-Digit Division Using Combinational Arithmetic Nets

CHIN TUNG, MEMBER, IEEE

Abstract—To meet the challenge created by the advent of large-scale integration, a unique microelectronic arithmetic building element and combinational arithmetic nets, composed of the building elements, have been studied and proposed for arithmetic processor design. A fast division algorithm, particularly suitable for floating-point arithmetic, has also been developed for signed-digit arithmetic. This algorithm is characterized by the need of preprocessing the divisor and then exact generation of quotient digits. This paper describes the implementation of this division algorithm with the arithmetic building element and combinational arithmetic nets. The intention here is to explore the feasibility of applying large-scale integration technology to arithmetic processors.

Index Terms—Arithmetic building element, combinational arithmetic net, division, large-scale integration, microelectronic block, signed-digit number system.

I. The Arithmetic Building Element and the Combinational Arithmetic Net

THE continuing reduction of the size and cost of microelectronic logic elements encourages the utilization of more complex logic nets in digital computers. In arithmetic processors, because of the application of these complex logic nets, the replacement of sequential logic nets by their combinational equivalents and the substitution of programmed software subroutines by hardware function generators are to be economically justified in the near future. A unique building block, the arithmetic building element (ABE), has been proposed [2], [3], [5], [7]. The ABE is defined in terms of arithmetic transfer functions, employing the redundant signed-digit (S-D) number system [1], in order to take full advantage of the complexity offered by microelectronics on a single chip. A combinational arithmetic (CA) net, composed of an acyclic array of the arithmetic building element, is organized in such a fashion that the accuracy, cost, and speed of approximating a given function are adjustable [4], [5], [7]. A CA net with feedback loops introduced is called an iterative CA net. As an economic alternative, hybrid CA nets can be formed with totally CA subnets and iterative CA subnets. A storage register at the ABE output facilitates "pipelining" on CA nets with resulting increase of computation throughput [4], [7]. A single ABE may also serve as the arithmetic processor of a serially organized computer. Without going into detail, we shall briefly describe only those characteristics of the ABE and the CA net which are pertinent to the implementation of the division algorithm with which this paper is concerned.

The simplified schematic diagram of the ABE is shown in Fig. 1. Let r be the radix of the S-D number system. The

Manuscript received September 18, 1969; revised February 12, 1970.

The author is with the IBM Research Division, San Jose, Calif.

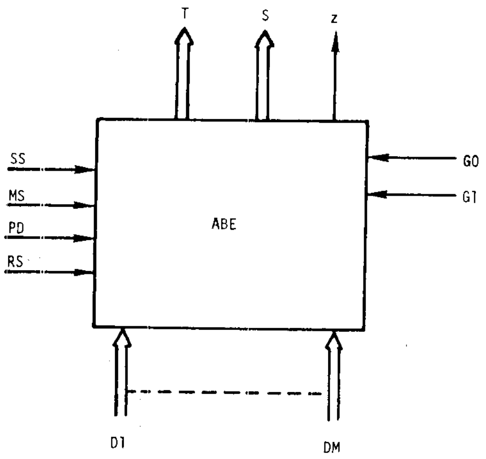

Fig. 1. ABE (simplified).

ABE can receive up to $(r+1)$ input digits $(X_i^1, x_i^2, \cdots, x_i^{r+1})$ on $(D1, D2, \cdots, DM)$ and produce one output digit s_i on S or two output digits (t_{i-1}, s_i) on (T, S). The function line inputs (SS, MS, PD, RS) are individual logic variables, allowing the selection of various transfer functions. Separate function line inputs $G0$ and $G1$ gate the output of the ABE which is held at zero only if $\sim G0 \vee G1 = 0$ where $\sim$ indicates negation. There is also a bit line output, z; $z=1$ when the result of the specified transfer function at the given digit position is zero. The arithmetic transfer functions pertinent to our discussion in this paper are described below.

A. Simple Sum

The ABE is activated to perform the simple sum functio with SS = 1. The digits of the two inputs words, x^1 and x^2 are x_i^1, x_i^2, x_{i+1}^1, and x_{i+1}^2. They are entered on input dig lines $D1$ to $D4$ in that order. The algorithm is

$$s_i = x_i^1 + x_i^2 - rg_i + g_{i+1} \qquad (1$$

where g_i is the carry generated at position i in an S-D num ber system and $g_i = 1$, 0, or -1. Thus, a single word s is th result of adding two words x^1 and x^2.

B. Multiple Sum

The ABE is activated to perform the multidigit sum fun tion with MS = 1. With up to m input digits $(x_i^1, \cdots, x_i^m)$ o $(D1, \cdots, DM)$ digit lines, the algorithm is

$$\sum_{j=1}^{m} x_i^j = rt_{i-1} + s_i \qquad ($$

where $m \le r+1$, $|s_i| \le a$, and $|t_{i-1}| \le a$. (a is the maximu digit value of a given S-D number system; $\lfloor (r/2)+1 \rfloor \le$ $\le r-1$.) At each digit position there are two output dig

, t_{i-1} and s_i; hence, a double word (s, t) is the result of ding m words.

Product

The ABE is activated to perform the product function th PD=1. The inputs are one digit x_j^1 (on $D1$) of the ltiplier x^1 and three digits $x_i^2, x_{i+1}^2, x_{i+2}^2$ (on $D2$, $D3$, $D4$) the multiplicand x^2. The algorithm is

$$\begin{cases} x_j^1 x_{i+2}^2 = & rp_{i+j+1} + q_{i+j+2} \\ x_j^1 x_{i+1}^2 = & rp_{i+j} + q_{i+j+1} \\ x_j^1 x_i^2 = & rp_{i+j-1} + q_{i+j} \end{cases} \quad (3)$$

$$S_{i+j} = p_{i+j} + q_{i+j} - rg_{i+j} + g_{i+j+1}. \quad (4)$$

e multiplication of two n-digit words, using this product gorithm, will yield a result where there are not fewer than digits in at least one digital position. Such a result is said be an n-tuple word. An n-tuple word can be reduced to a ngle word by using more ABE's which perform MS and S functions on the n-tuple word.

Reconversion

The ABE is activated to reconvert an S-D number to inary form with RS=1. A CA net is an ensemble of ABE's.

The graphical representation of CA nets, in the form of irected graphs, exists in two levels, namely, the algorithm nd hardware levels [5]. We shall describe the hardware vel only. The emphasis on graphical representation of CA ets at this level is for the purpose of exhibiting functional roperties and limitations of the ABE. The vertices which epresent the four arithmetic transfer functions just mentioned are:

1) simple sum: $\oplus$.
2) multiple sum: $\bigoplus$.
3) product: $\circledast$.
4) reconversion: (RS).

Further, the ABE also provides storage capability; hence,
5) storage: Ⓢ.

Arcs are represented by directed solid lines. Digits transmitted and their formats are identified along arcs; left shift ($k\updownarrow$) or right shift ($k\updownarrow$) k places can also be indicated if necessary. If no such information is explicitly indicated, then the whole word, as it was produced in the vertex, is to be transmitted without shift. The setting of control signals, if any, is also indicated along the arcs. The special outputs, e.g., z, from the preceding vertex assume the form of encoded digits 0 or 1 if they stand by themselves along the arc. However, if they are followed by "$|G0$" or "$|G1$", then they assume the form of a single bit 0 or 1 and are fed into the succeeding vertex to energize the gating control $G1$ or $G0$. Complementation may be specified for any input word by putting a solid dot at the termination of the incoming arc transmitting the input word.

The delay time through one ABE is the same as that through one hardware level vertex and is defined as one time unit (tu).

II. The Signed-Digit Division Algorithm

S-D number representation which is a redundant and positional representation with a constant radix $r \geq 3$ allows arithmetic addition with carry propagation limited to one digit position regardless of the length of the operands [1].

A division algorithm characterized by exact generation of quotients has been developed for S-D arithmetic [8]. The basic principle is outlined below. All numbers are assumed to have the format

$$x = (x_{-m} \cdots x_{-1} x_0 \cdot x_1 \cdots x_n). \quad (5)$$

It is further assumed that the divisor d is positive and has the format

$$d = d_0 \cdot d_1 \cdots d_n. \quad (6)$$

The recursive step in division is described by

$$x^{(j+1)} = rx^{(j)} - q_{j+1}d, \qquad j = 0, 1, \cdots, n-1 \quad (7)$$

where $x^{(0)}$ is the dividend, $x^{(j)}$ is the jth partial remainder, q_{j+1} is the $(j+1)$st digit of the quotient q, and r is the radix. If it is desired to have the quotient digits generated in an exact manner (i.e., not trial-and-error) and, in particular, to have q_{j-1} equal to $x_1^{(j)}$, plus a constant (which may be zero), then this requirement together with the constraint that $|x^{(j+1)}|$ be less than d would yield a specific range for d. For a minimally redundant S-D number system it has been found [8] that

$$1 - h_1 < d < 1 + h_2$$

where for even radices r,

$$\begin{aligned} h_2 &= (r-4)/r(r-1) \\ h_1 &= (r-4)/(r-1)(r+4), \end{aligned} \quad (8)$$

for odd radices r,

$$\begin{aligned} h_2 &= (r-3)/(r-1)^2 \\ h_1 &= (r-3)/(r-1)(r+3). \end{aligned}$$

It should be pointed out that ranges for divisors in conventional number systems have also been investigated [6].

Based on the above argument this division algorithm is performed in two steps. In step one, the input divisor, assumed positive and normalized, is brought into the specified range by no more than two multiplications. A nonoptimal set of multiplication constants has been developed [8], and the selection of the constants is based on the examination of the unit digit and the two most significant digits of the fraction of the divisor. In the meantime the dividend is adjusted accordingly. At the end of step one, $d = 1 + e$, $e = (0 \cdot e_1 \cdots e_n)$, and e satisfies the conditions in (8), $x = (x_{-1} x_0 \cdot x_1 \cdots x_n)$.

The recursive relation in step 2 can be described with the following formula, analogous to (7):

$$x^{(j+1)} = rx^{(j)} - (t_0^{(j)} r + x_1^{(j)})e \qquad j = 0, 1, \cdots, n-1 \quad (9)$$

and

$$q_{j+1} = x_0^{(j+1)}$$

where $t_0^{(j)}$ is the carry digit out of the most significant digit of the fraction of the partial remainder. A formal description

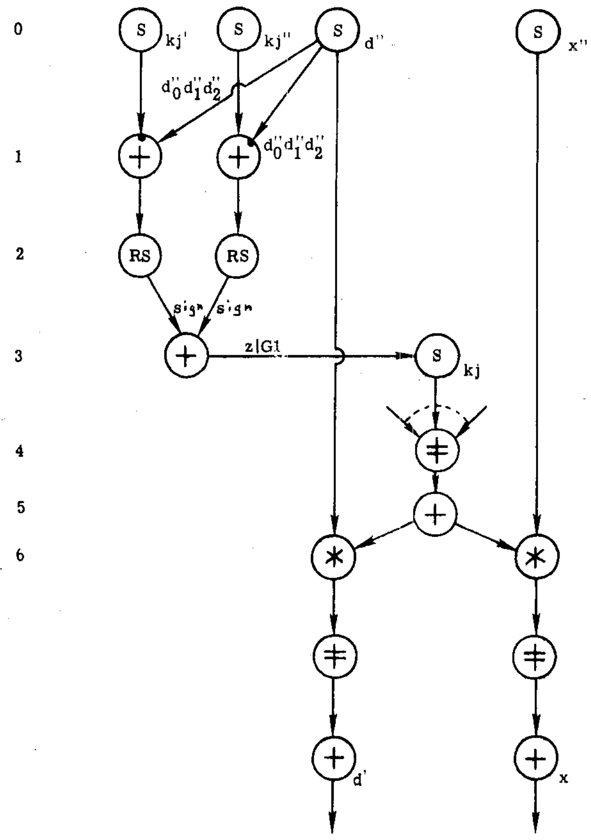

Fig. 2. CA net of step one.

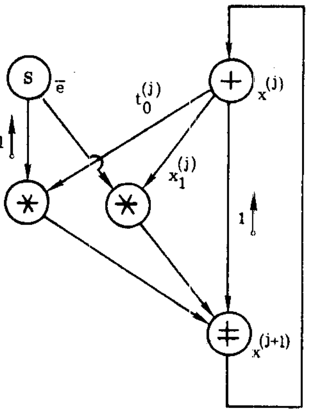

Fig. 3. CA net of step two.

of step 2 using Iverson's notation and an example of radix 16 with minimum redundancy can be found in [8].

III. Implementation

The basic operations in step one of this S-D division algorithm are as follows. Compare the given argument against a set of constants and then select, based on the result of the comparison, an appropriate multiplication factor by which to multiply the given argument. The quantities (x'', d'') are the given dividend and divisor, respectively, with d'' assumed to be positive and normalized. In the first substep $x' = x''k_j$ and $d' = d''k_j$ if $k_j' \leq d'' \leq k_j''$, where (k_j', k_j'') is a pair of comparison constants and k_j is the associated multiplication factor. In the second substep the above process is repeated; $x = x'k_j$ and $d = d'k_j$, if $k_j' \leq d' \leq k_j''$. A representative segment of a possible CA net implementation of the above process is shown in Fig. 2.

To test whether $a \leq y \leq b$ is equivalent to testing whether $(y-a) \geq 0$ and $(b-y) \geq 0$. This is exactly the way it is determined whether or not $d_0' \cdot d_1''d_2''$ is within the range bounded by k_j' and k_j'' in Fig. 2. First, the two subtractions are performed. The subsequent reconversions yield the sign (polarity) information of the differences. If and only if the two differences are nonnegative, the sum of their signs will yield the indication $z = 1$. The zero indication is then used as the gating control $G1$ to release the multiplication factor k_j. Of all multiplication factors, only one will be released and the merging operation [5] is done at the multiple sum vertex at level 4. The CA net described so far has performed the functions of comparison, decision, and selection. The appropriately selected multiplication factor k_j is then used to multiply d'' and x'' to obtain d' and x'. The above process may be repeated to obtain d and x.

The inputs to step two are $d = 1 + e$ and x. The CA net describing the recursive relation is shown in Fig. 3. This step is relatively more straightforward than step one. Note that $-t_0^{(j)}re$ is formed by shifting $\bar{e}$ to the left by one digit position and then multiplying it by $t_0^{(j)}$. The quantity $rx^{(j)}$ is obtained also by shifting $x^{(j)}$ to the left by one digit position. Thus, the input to the multiple sum vertex is the quantity

$$(r\bar{e}t_0^{(j)} + x_1^{(j)}\bar{e} + rx^{(j)})$$

which is $x^{(j+1)}$. Note that there should also exist a storage vertex receiving the quotient digits generated.

IV. Conclusion

The implementation of an S-D division algorithm characterized by exact generation of quotients with a unique microelectronic arithmetic building block has been shown. We have also demonstrated the decision and selection capability of the combinational arithmetic net composed of the building blocks. We by no means attempt to suggest that this scheme is better or more efficient than current techniques employed in existing arithmetic processors. The above discussion, however, does motivate our interest in exploring a new approach to arithmetic processor design for the forthcoming era of large-scale integration.

References

[1] A. Avizienis, "Signed-digit number representations for fast parallel arithmetic," *IRE Trans. Electronic Computers*, vol. EC-10, pp. 389–400, September 1961.
[2] ——, "Arithmetic microsystems for the synthesis of function generators," *Proc. IEEE*, vol. 54, pp. 1910–1919, December 1966.
[3] A. Avizienis and C. Tung, "Design of combinational arithmetic nets," *Dig. 1st Ann. IEEE Computer Conf.* (Chicago, Ill.), pp. 25–28, September 1967.
[4] C. Tung and A. Avizienis, "Combinational arithmetic systems for the approximation of functions," *1970 Spring Joint Computer Conf., AFIPS Proc.*, vol. 36. Washington, D. C.: Spartan, 1970, pp. 95–107.
[5] A. Avizienis and C. Tung, "A universal arithmetic building element (ABE) and design methods for arithmetic processors," this issue, pp. 733–745.
[6] S. K. Nandi and E. V. Krishnamurthy, "A simple technique for digital division," *Commun. ACM*, vol. 10, pp. 299–301, May 1967.
[7] C. Tung, "A combinational arithmetic function generation system," Ph.D. dissertation, Department of Engineering, University of California, Los Angeles, Rept. 68-29, June 1968.
[8] ——, "A division algorithm for signed-digit arithmetic," *IEEE Trans. Computers* (Short Notes), vol. C-17, pp. 887–889, September 1968.