CONSTANT NUMBER PARALLEL MULTIPLIERS

J. Sifnaios Chr. Meletis P. Bougas P. Kalivas K. Pekmestzi

e-mail:(john,chris,paul,paraskevas,pekmes)@microlab.ntua.gr

Department of Electrical and Computer Engineering, National Technical University of Athens, 157 73 Zographou, Athens, Greece pekmes@microlab.ntua.gr.

Key words: Parallel multipliers, Constant number multipliers, Canonic signed digit representation

ABSTRACT

A parallel multiplier for constant numbers is presented. The constant number is in Canonical Signed Digit (CSD) form and the other factor in two's complement form. The result is obtained in two's complement form. The design presented here is based on a special algorithm developed for the multiplication of two's complement numbers with numbers in signed digit representation. The proposed multiplier compared with the existing schemes is superior from the point of hardware complexity and operation speed.

I. INTRODUCTION

An important issue in the design of digital systems, such as digital filters, is the hardware saving that can be achieved, when the coefficients of the involved multiplications are constant numbers. Apparently, the knowledge of the coefficient at the design phase allows a significant decrease in hardware by eliminating the multiplier cells corresponding to zero coefficient bits. Furthermore, hardware reduction can be achieved by using special representations like the modified Booth's coding [1], which reduces the number of the coefficient bits to the half. The Canonical Signed Digit (CSD) representation [2] gives better results in hardware reduction because it minimizes the non-zero bits of the coefficient [3].

The CSD representation has been extensively used in the design of digital filters without multipliers [4]. The coefficients represented in CSD form are approximated using two or three non-zero digits. However, a systematic approach for the implementation of a parallel multiplier with CSD numbers is useful. This approach can be used not only for implementing digital filters but also for FFT and cryptography applications.

An algorithm for the multiplication of two's complement numbers with numbers in CSD form has been presented in [5], [6]. in this paper, we adapt this algorithm and present a new parallel multiplier of two's complement numbers with a constant factor in CSD form. Specifically, in Section II the multiplication algorithm is comprehensively presented. The detailed design of the parallel multiplier is given in Section III. Also, a comparison between this multiplier and the parallel multiplier with the constant factor in two's complement form and modified Booth's form is given.

II. DESCRIPTION OF THE MULTIPLICATION ALGORITHM

Let us consider the multiplication of a two's complement number X

$$X = -x_{n-1} \cdot 2^{n-1} + \sum_{j=0}^{n-2} x_j \cdot 2^j \tag{1}$$

with a constant coefficient a in signed digit form, which is

$$a = \sum_{i=0}^{m-1} a_i \cdot 2^i, \ a_i = 0,\pm 1 . \tag{2}$$

The multiplication is the expression

$$P = X \cdot a = \sum_{i=0}^{m-1} P_i \cdot 2^i \tag{3}$$

where $P_i = X \cdot a_i = -x_{n-1} \cdot a_i \cdot 2^{n-1} + \sum_{j=0}^{n-2} x_j \cdot a_i \cdot 2^j$ (4)

According to the algorithm the final product is

$$P = -2^{n+m-1} + \sum_{i=0}^{m-1} \overline{a}_i \cdot 2^{n+i-1} + 2^{n-1} + S + \sum_{i=0}^{m-1} P_i^* \cdot 2^i \tag{5}$$

where

$$P_i^* = (\overline{x}_{n-1} \oplus s_i) \cdot a_i \cdot 2^{n-1} + \sum_{j=0}^{n-2} (x_j \oplus s_i) \cdot a_i \cdot 2^j \quad \text{and}$$

$$S = \sum_{i=0}^{m-1} s_i 2^i \tag{6}$$

In the above equations a_i and s_i represent the absolute value and the sign of a_i digit respectively. Specifically, the value of s_i is assumed to be 1 for $a_i < 0$ and 0 otherwise. The symbol $\oplus$ represents the XOR operator. The results of the inversion and of the XOR operator are used as arithmetic quantities in the arithmetic expressions. In (6) the quantity $x_j \oplus s_i$ implies the inversion of x_j, if a_i

is negative and the factor a_i expresses the partial product elimination, if a_i is zero.

Equation (5) implies that for the implementation of the above two's complement multiplication, the number $\bar{x}_{n-1}x_{n-2}\cdots x_1x_0$ properly weighted must be added inverted or not if a_i is positive or negative respectively. In addition, the correction term

$$C=-2^{n+m-1}+\sum_{i=0}^{m-1}\bar{a}_i\cdot 2^{n-1+i}+2^{n-1}+S \tag{12}$$

that depends only on a, must be included.

II. IMPLEMENTATION OF THE PARALLEL MULTIPLIER

In fig. 1 it is shown how the proposed multiplier for $a=10\bar{1}00101$ and $n=m=8$ is deduced from a general array multiplier by eliminating the partial product rows that correspond to non-zero coefficient digits. In the rows, to which a negative coefficient bit corresponds, the bits of *X* are inverted except from the most significant bit, which is inverted in the rows, which correspond to a positive coefficient bit. Apparently, no Full-Adders is required for the two first partial product rows. The final circuit is shown in Fig. 2. It outputs the result in carry-save form and consequently a final adder is required for the result to be obtained in two's complement from.

Also, the circuit of Fig. 2 adds the correction term *C* given by (12). For convenience, we consider this term as two parts, the upper part

$$C_{Up}=-2^{n+m-1}+\sum_{i=0}^{m-1}\bar{a}_i\cdot 2^{n-1+i}+2^{n-1} \tag{13}$$

and the lower part

$$C_{Lo}=S \tag{14}$$

For $a=10\bar{1}00101$ we have $C_{Up}=0101101$ and C_{Lo}=001000. The bits of C_{Up} and C_{Lo} are shown in bold in Fig. 2. Considering the correspondence of bits of C_{Up} in this figure, we see that always the steps shaped by overlapped partial product rows in the left margin of the multiplier are always covered by '1's except from the edge of each step where a zero bit corresponds. This fact allows the addition of '1's by just inverting and extending leftwards one input of the adder in the right end of the step as shown in Fig. 1.

The term -2^{n+m-1} is implemented by inverting the carry output of the final adder.

The first partial product row because of term 2^{n-1} is covered by zeros except the left end of this row where a '1' corresponds.

As far as the addition of C_{Lo} is concerned, we exploit the fact that a '1' must be added to the right end of each partial product row if the coefficient bit for this row is negative. However, there is no need for extra adders because these bits can be incorporated in the lower part of the final adder of the multiplier. In all cases there is place for this addition except from the right end of the second row. In most applications, like digital filters the lower part of the result is truncated, so there is no need for these '1's to be added. Even in the case where the whole prod-

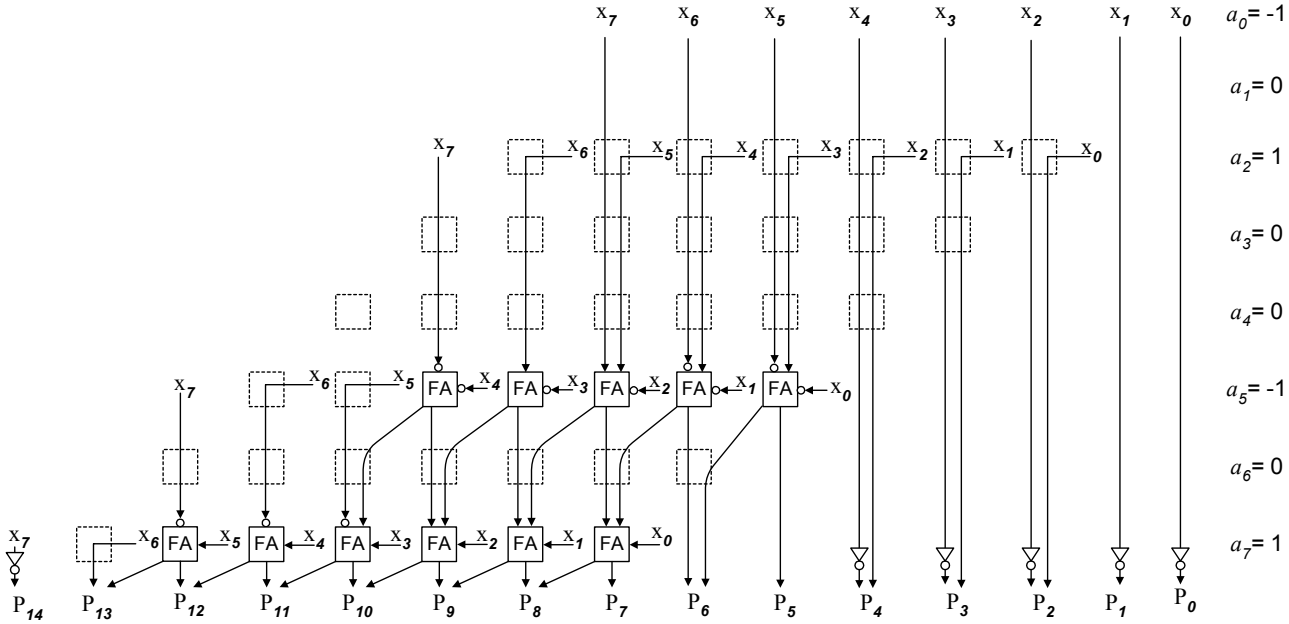

Figure 1. A parallel multiplier with eliminated the rows corresponding to the zero bits of the coefficient $a=10\bar{1}00101$.

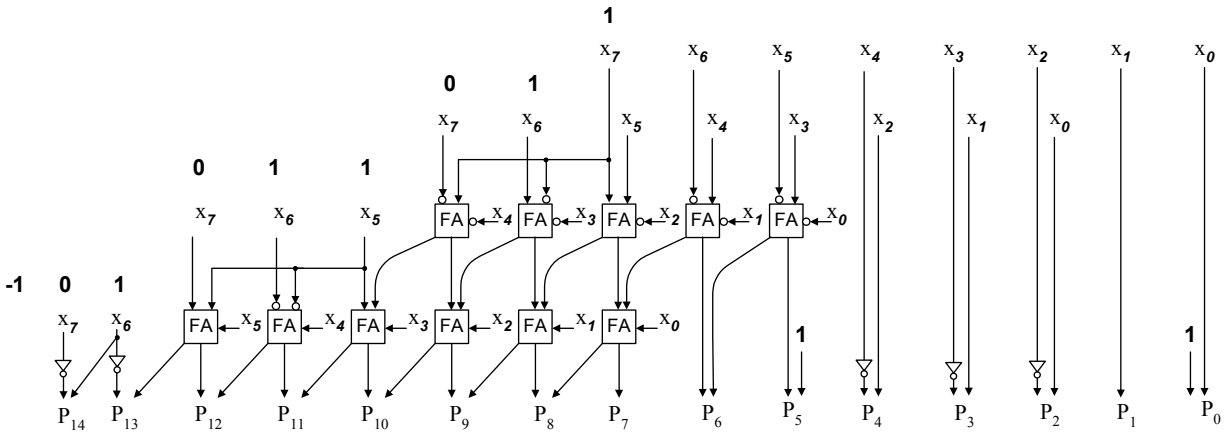

Figure 2. A parallel multiplier for the coefficient $a = 10\bar{1}0010\bar{1}$ and 8-bit word length.

product is required the above difficulty can be easily overcome by modifying properly the final adder.

The required hardware for the above scheme is directly proportional to the number of non-zero bits in the constant coefficient. Furthermore, it depends on the exact position of these bits inside the coefficient. Thus, if k is the number of non-zero bits the multiplier will consist of k partial product rows and only $k-2$ of them will have to be added using Full-Adders. The number of Full-Adders required by each of these $k-2$ rows depends on the relative position of the non-zero bits inside the coefficient.

Particularly, if a_i is the coefficient bit for a row then this line will include $m-d-1$ adders where d is the number of zeros between a_i and the next lower order non-zero coefficient bit. The combinational delay is $(k-2)*D_{FA}$ where D_{FA} is the delay of a Full-Adder.

The hardware complexity of the proposed architecture is given in Table I. For the computations we have assumed the average case, where the constant number of length m consists of $\frac{m}{3}$ non-zero bits. We have also assumed that these bits are evenly distributed inside the coefficient, namely they are separated by two zero bits $(d=2)$. For comparison, the hardware complexity of a parallel multiplier, when the constant coefficient is represented in two's complement form is also given in Table I. For the computations, we assumed the average case, where the half bits of the constant coefficient are zeros and evenly distributed inside the coefficient. In both cases the hardware of the final adder has been not taken into account. The comparison reveals that the hardware complexity and the combinational delay of the proposed scheme is about 40% decreased compared with the complexity and the delay when the constant coefficient is represented in two's complement form and 20% compared with the multiplier where the constant coefficient is in modified Booth's form.

TABLE I

HARDWARE COMPLEXITY AND COMBINATIONAL DELAY OF THE PROPOSED MULTIPLIERS AND THE PARALLEL MULTIPLIER FOR CONSTANT COEFFICIENT IN TWO'S COMPLEMENT FORM

Type of multiplier	Hardware Complexity (Average)	Combinational Delay
Parallel multiplier for constant coefficient in two's complement form	$(\frac{m}{2}-2)\cdot(m-2)\cdot FA$	$(\frac{m}{2}-2)\cdot D_{FA}$
Parallel multiplier for constant coefficient in Modified Booth's form	$(\frac{3m}{8}-2)\cdot(m-2)\cdot FA$	$(\frac{3m}{8}-2)\cdot D_{FA}$
Proposed parallel pipeline multiplier	$(\frac{m}{3}-2)\cdot(m-3)\cdot FA$	$(\frac{m}{3}-2)\cdot D_{FA}$

D_{FA}: Full-Adder delay

V. CONCLUSION

We proposed the design of a parallel multiplier where the one factor is constant and represented in canonical signed digit form. The hardware and the combinational delay are reduced significantly compared with a parallel scheme where the constant factor is represented in two's complement form. Moreover, canonic structure of a parallel multiplier is preserved and thus, this scheme can be described as a parameterized component in a VHDL library with the multiplier coefficient as parameter.

The presented design has been carried out in detail and verified by extensive simulations for various coefficients. The proposed multiplier's design is part of an FFT computation circuits.

REFERENCES

1. A. Booth, A signed binary multiplication technique, Quart. J. Mech. Appl. Math., vol. 4, pp. 236-240, 1951.
2. G. Reitwiesner, Binary Arithmetic, Advances in Computers, vol. 1, pp. 231-308, 1966.
3. A. Peled, On the hardware implementation of digital signal processors, IEEE Trans. Acoust., Speech, Signal Proc., vol. 24, 1976, pp. 76-86.
4. T. Lin, H. Samueli A 200-MHz CMOS x/sin(x) digital filter for compensating D/A converter frequency response distortion, IEEE J. of Solid-State Circuits, vol. 26, no 9, pp. 1278-1285, 1991.
5. K. Z. Pekmestzi, P. Kalivas Constant number serial pipeline multipliers, Kluwer Academic Publishers, The Journal of VLSI Signal Processing, Volume 26, Issue 3, November 2000, pp. 361-368.
6. K. Z. Pekmestzi, P. Kalivas, Constant Number Multipliers, The 5th International Symposium on Bases of Electronics (SBE'99), Romania 1999.