A SPT based Low Complexity Realization of the Weight Update Loop of an Adaptive Filter

Sunav Choudhary, Pritam Mukherjee, Mrityunjoy Chakraborty*
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur, INDIA
E.Mail : mrityun@ece.iitkgp.ernet.in

***Abstract*—The "sum of power of two (SPT)" is an effective format to represent multipliers in a digital filter which reduces the complexity of multiplication to a few shift and add operations. The canonic SPT is a special sparse SPT representation that guarantees occurrence of at least one zero between every two non-zero SPT digits. In the case of adaptive filters, as the coefficients are updated with time continuously, no fixed SPT representation of the filter weights is, however, possible. Conversion of the filter coefficients into canonic SPT form at each time index is also impractical. To overcome this problem, this paper presents a bit serial algorithm for carrying out the weight updating of an adaptive filter in the canonic SPT domain only. For this, a novel algorithm is presented for bit serial addition of two numbers, one given in canonic SPT form and the other in 2's complement form, to produce a result in canonic SPT. The proposed algorithm uses the properties of canonic SPT numbers effectively, resulting in considerable reduction of the hardware complexity of the bit serial adder.**

I. INTRODUCTION

In a digital filter, the complexity of realization both in terms of silicon area and time is determined primarily by the multipliers. Consequently, efforts have been made to design filters that are free of multipliers. Ideally, if a multiplier is replaced by a single signed power of two term, the complexity reduces enormously since multiplication by a power of two amounts to a simple shift operation. However, the coefficient quantization error in such cases can be substantial affecting the filter performance considerably. A more effective approach for this is to approximate each multiplier by a sum of (signed) power of two (SPT) while keeping the number of power of two terms as few as possible. A well known sparse SPT representation in this context is the so-called canonic SPT [1]. Under this, a coefficient, say w, is represented as,

$$w = \sum_{r=1}^{R} s(r)\, 2^{g(r)}, \tag{1}$$

where $s(r) \in \{1, -1, 0\}$ is the r-th SPT coefficient (also called digit in this paper), $g(r)$ is an increasing sequence of integers and R is the number of terms specified a priori. In canonic SPT, no two consecutive terms are non-zero (i.e., ± 1) simultaneously, i.e., if for any r, $s(r) = \pm 1$, then both $s(r+1$ and $s(r-1)$ must be zero (for example, $11 = 2^4 - 2^2 - 2^0$, $19 = 2^4 + 2^2 - 2^0$ etc.). In other words, the canonic SPT guarantees that at least $\lfloor \frac{R}{2} \rfloor$ SPT coefficients in (1) are zero[1]. A circuit to convert 2's complement numbers into canonic SPT, both in bit serial and parallel forms, has been presented in [1].

The SPT format has been used widely by researchers over years for efficient realization of fixed coefficient digital filters ([4]-[10]). The proposed algorithms are, however, offline techniques which can not be used for realizing adaptive filters whose coefficients change with time and thus can not be represented by a fixed SPT expression. Conversion of the filter weights into the canonic SPT form at each time index is also not practical from hardware complexity point of view. Note that in an adaptive filter (e.g., the LMS algorithm), the filter weights are updated as,

$$Future\_Weight = Current\_Weight + Update. \tag{2}$$

[1]Alternatively, one can view the canonic SPT as a special case of hybrid-2, redundant arithmetic [3] based representation where no two successive bits are allowed to be non-zero simultaneously

Proceedings of the Second APSIPA Annual Summit and Conference, pages 347–349,
Biopolis, Singapore, 14-17 December 2010.

Assuming that the $Current\_Weight$ is available in canonic SPT form and the $Update$ term which is a function of the input is given in 2's complement, a more effective approach would be to carry out the above weight updating in the canonic SPT domain only, generating the $Future\_Weight$ in canonic SPT form. Towards this objective, we present a technique in this paper for bit serial addition of two numbers, one given in canonic SPT form and the other in 2's complement, to produce the result in canonic SPT.

II. Proposed Algorithm for SPT Addition

Given a $(N+1)$ digit canonic SPT number $a = a_N a_{N-1} \cdots a_1 a_0$ $(\equiv \sum_{i=0}^{N} a_i\, 2^i)$, $a_i \in \{1, -1, 0\}$, $i = 0, 1, \cdots, N$, and a $(n+1)$ bit 2's complement number $b = b_N b_{N-1} \cdots b_1 b_0$ $(\equiv -b_N 2^N + \sum_{j=0}^{N-1} b_j\, 2^j)$, $b_j \in \{1, 0\}$, $j = 0, 1, \cdots, N$, first we note that the number b can be interpreted as a SPT number (not necessarily canonic), with the bit b_j for $j = 0, 1, \cdots, N-1$ being actually a SPT digit taking values of 0 or 1 (for the sign bit b_N, it is a SPT digit taking values of -1 or 0). In the proposed scheme, in the i-th cycle, we add a_i, b_i and the incoming carry c_i generated in the $(i-1)$-th cycle ($c_0 = 0$) and produce the new carry c_{i+1} and an intermediate result sp_i, which is to be adjusted to the final value s_i in the $(i+1)$ clock cycle. In other words, in the proposed scheme, there is a latency of one cycle between the i-th cycle input and the corresponding output. The proposed algorithm is given below where we use the notation 1^* to denote ± 1.

Algorithm : *Given a_i, b_i, c_i and sp_{i-1}, carry out the following steps at the i-th cycle :*

Step 1 (Addition) : Add a_i, b_i and c_i to produce c_{i+1} and sp_i.

Step 2 (Adjustment) : For adjustment, we utilize the following identities : $2^i + 2^{i-1} = 2^{i+1} - 2^{i-1}$ and $2^i - 2^{i-1} = 2^{i-1}$.

• *If $sp_i = 1^*$ and $sp_{i-1} = -1^*$, then adjust sp_i to 0 and take $s_{i-1} = 1^*$ as the output (of the previous cycle).*

• *If $sp_i = sp_{i-1} = 1^*$, then take $s_{i-1} = -1^*$, adjust sp_i to 0 and propagate 1^* to the $(i+1)$-th step as c_{i+1}* [Note that for this case, c_{i+1} from Step 1 can not be 1^*, since this would imply that all the three digits, a_i, b_i and c_i are 1^* each simultaneously, which along with $sp_{i-1} = 1^*$ is, however, not possible, as per Lemma 1 below].

• No adjustment needed otherwise, meaning $sp_{i-1} \rightarrow s_{i-1}$.

As seen above, the four digits, a_i, b_i, c_i and sp_{i-1} are used to generate s_{i-1} and c_{i+1}. Theoretically, these four digits can have a total of $3 \times 2 \times 3 \times 3 = 54$ combinations. However, as shown by Lemmas 1-2 below, only a fraction of these combinations are feasible while the remaining ones can not come up. *This results in considerable savings in hardware as one can use the so-called "don't care" states for the invalid combinations.*

Lemma 1 : *The three digits, a_i, c_i and sp_{i-1} can not be non-zero simultaneously.*

[The proof of this Lemma is omitted in this paper].

Lemma 2 : *The digit c_i can not take the value of -1.*

[The proof of this Lemma is omitted in this paper].

III. Circuit Implementation

As seen above, the algorithm is simply a rule to transform the pair $(c_i + a_i + b_i,\ sp_{i-1})$ to the triplet (c_{i+1}, sp_i, s_{i-1}). Define three functions, $f_c(a_i, b_i, c_i, sp_{i-1}), f_{sp}(a_i, b_i, c_i, sp_{i-1})$ and $f_s(a_i, b_i, c_i, sp_{i-1})$ which generate the quantities c_{i+1}, sp_i and s_{i-1} respectively at the i-th cycle following the above algorithm. The truth table for each function is easy to generate. However, out of a total $3 \times 2 \times 3 \times 3 = 54$ possible combinations in each truth table, *the Lemmas 1-2 can be used to reduce the number of combinations to just 26.*

The corresponding block diagram for hardware realization of the bit serial adder is shown in Fig. 1. Here f_c, f_{sp} and f_s are combinatorial blocks which implement the respective truth tables. Note that since the SPT representation uses altogether three values, namely, 1, -1, 0, in a digital implementation, each SPT digit a_i is represented by two binary bits, as per the following : $(1)_{SPT} = (01)_2$, $(0)_{SPT} = (00)_2$ and $(\bar{1})_{SPT} = (11)_2$, which is consistent with 2's complement representation of signed binary numbers. In the case b, for

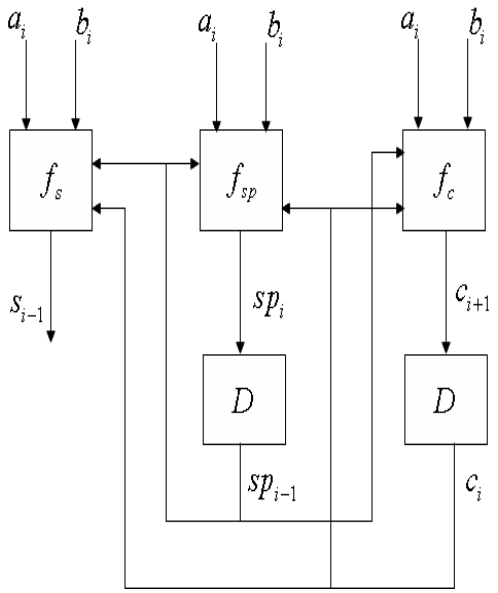

Fig. 1. Block Diagram for Bit-Serial Addition of a Canonic SPT Number a with a 2's Complement Number b, producing the output in Canonic SPT form.

$i = 0, 1, \cdots, N-1$, the digit b_i will be represented by the 2 bit binary number $0\, b_i$, whereas b_N will be represented by $b_N\, b_N$ (i.e., 1 bit extension).

IV. Acknowledgement

This work has been supported partially by a research grant from the ministry of information technology, government of India.

References

[1] Lim Y.C, Evans J.B. and Liu B., "Decomposition of Binary Integers into Signed Power-of-Two Terms", *IEEE Trans. Circuits and Systems*, 1991, **38**, (6), pp. 667-672.

[2] Lim Y.C., Yang R., Li D. and Song J., "Signed power-of-two term allocation scheme for the design of digital filters", *IEEE Trans. Circuits and Systems, part II*, 1999, **46**, (5), pp. 577-584.

[3] Parhi K.K., "VLSI Digital Signal Processing Systems - Design and Implem,entation" (John Wiley and Sons Inc., 1999)

[4] Dam H.H., Cantoni A., Teo K.L. and Nordholm S., "FIR Variable Digital Filter With Signed Power-of-Two Coefficients", *IEEE Trans. Circuits and Systems, Part I*, 2007, **54**, (6), pp. 1348-1357.

[5] Park S.Y., Cho N.I., "Design of Multiplierless Lattice QMF: Structure and Algorithm Development", IEEE Trans. Circuits and Systems, part II, 2008, **55**, (2), pp. 173-177.

[6] Feng Z. G. Feng, Teo K.L., "A Discrete Filled Function Method for the Design of FIR Filters With Signed-Powers-of-Two Coefficients", *IEEE Trans. Signal Processing*, 2008, **56**, (1), pp. 134-139.

[7] Aktan M., Yurdakul A. and Dundar G., "An Algorithm for the Design of Low-Power Hardware-Efficient FIR Filters", *IEEE Trans. Circuits and Systems part I*, 2008, **55**, (6), pp. 1536-1545.

[8] Yu Y.J. and Lim Y.C., "Design of Linear Phase FIR Filters in Subexpression Space Using Mixed Integer Linear Programming", *IEEE Trans. Circuits and Systems part I*, 2007, **54**, (10), pp. 2330-2338.

[9] Xu F., Chang C.H. and Jong C.C., "Design of Low-Complexity FIR Filters Based on Signed-Powers-of-Two Coefficients With Reusable Common Subexpressions", *IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems*, 2007, **26**, (10), pp. 1898-1907.

[10] Park S.Y. and Cho N.I., "Design of signed powers-of-two coefficient perfect reconstruction QMF Bank using CORDIC algorithms", *IEEE Trans. Circuits and Systems part I*, 2007, **53**, (6), pp. 1254-1264.