Implementation of Memory Based Multiplication Using Micro wind Software

U.Palani 1, M.Sujith2, P.Pugazhendiran3

1IFET College of Engineering, Department of Information Technology, Villupuram
2,3IFET College of Engineering, Department of Electrical and Electronics Engineering, Villupuram
Email:2Sujijet@yahoo.in, 3pugazhifet@gmail.com

Abstract: The antisymmetric product coding techniques for lookup-table design for memory-based multipliers used in digital circuits. By using this techniques it results in reduction of the LUT. We present a different form of APC for efficient optimization of memory based applications. The proposed combined approach provides a reduction in size compared to the conventional LUT. It is shown that the proposed microwind-DShC based LUT for small sizes can be used for efficient implementation of memory based processing. It is found that the proposed LUT-based multiplier shows compact area and time complexity for a word size of 8 bits significantly less multiplication time. The model of LUT based multiplier is designed and executed using micro wind and DSCH tools.

Index terms: Anti symmetric Product coding, Look Up Table, System on Chip.

I. INTRODUCTION

Along with the progressive device scaling, semiconductor memory has become cheaper, faster, and more power-efficient. Moreover, according to the projections of the international technology roadmap for semiconductors [1], embedded memories will have dominating presence in the system on-chips (SoCs), which may exceed 90%, of the total SoC content. It has also been found that the transistor packing density of memory components is not only higher but also increasing much faster than those of logic components. Apart from that, memory-based computing structures are more regular than the multiply–accumulate structures and offer many other advantages, e.g., greater potential for high-throughput and low-latency implementation and less dynamic power consumption. Memory-based computing is well suited for many digital signal processing (DSP) algorithms, which involve multiplication with a fixed set of coefficients. A conventional lookup-table (LUT)-based multiplier is shown in Fig. 1, where A is a fixed coefficient, and X is an input word to be multiplied with A.

Fig. 1. Conventional LUT-based multiplier.

Assuming X to be a positive binary number of word length L, there can be 2L possible values of X, and accordingly, there can be 2L possible values of product \( C = A \cdot X \). Therefore, for memory-based multiplication, an LUT of 2L words, consisting of precomputed product values corresponding to all possible values of X, is conventionally used. The product word \( A \cdot X_i \) is stored at the location \( X_i \) for \( 0 \leq X_i \leq 2^L - 1 \), such that if an L-bit binary value of \( X_i \) is used as the address for the LUT, then the corresponding product value \( A \cdot X_i \) is available as its output. Several architectures have been reported in the literature for memory-based implementation of DSP algorithms involving orthogonal transforms and digital filters [2]–[5]. Recently, we have presented a new approach to LUT design, where only the odd multiples of the fixed coefficient are required to be stored [5], which we have referred to as the odd-multiple-storage (OMS) scheme in this brief. In addition, we have shown that, by the antisymmetric product coding (APC) approach, the LUT size can also be reduced to half, where the product words are recoded as antisymmetric pairs [10]. However, the OMS technique in [3] cannot be combined with the APC scheme, since the APC words generated according to [4] are odd numbers. Moreover, the OMS scheme in [5] does not provide an efficient implementation when combined with the APC technique. In this brief, we therefore present a different form of APC and combined that with a modified form of the OMS scheme for efficient
memory based multiplication. In the next section, we have discussed the modified APC and the combined OMS–APC approach. The implementation of combined OMS–APC scheme is described in Section III. The synthesis results of the proposed multiplier and canonical-signed-digit (CSD)-based multipliers, along with the conclusion, are presented in Section IV.

**II. PROPOSED LUT OPTIMIZATIONS FOR MEMORY-BASED MULTIPLICATION**

**A. APC for LUT Optimization**

For simplicity of presentation, we assume both X and A to be positive integers. The product words for different values of X for L = 5 are shown in Table I. It may be observed in this table that the input word X on the first column of each row is the two’s complement of that on the third column of the same row. In addition, the sum of product values corresponding to these two input values on the same row is 32A. Let the product values on the second and fourth columns of a row be u and v, respectively. Since one can write \( u = [(u + v)/2] - (v - u)/2 \) and \( v = [(u + v)/2 + (v - u)/2] \), for \( (u + v) = 32A \), we can have

\[
u = 16A - [(v - u)/2] \quad v = 16A + [(v - u)/2] \tag{1}\]

The product values on the second and fourth columns of Table I therefore have a negative mirror symmetry. This behavior of the product words can be used to reduce the LUT size, where, instead of storing u and v, only \( [(v - u)/2] \) is stored for a pair of input on a given row. The 4-bit LUT addresses and corresponding coded words are listed on the fifth and sixth columns of the table, respectively. Since the representation of the product is derived from the antisymmetric behavior of the products, we can name it as antisymmetric product code. The 4-bit address \( X' = (x3x2x1x0) \) of the APC word is given by

\[
x = \begin{cases} X, & \text{if } x_4 = 1 \\ X', & \text{if } x_4 = 0 \end{cases} \tag{2}\]

where \( XL = (x3x2x1x0) \) is the four less significant bits of X, and \( XL \) the desired product could be obtained by adding or subtracting the stored value \( (v - u) \) to or from the fixed value 16A when \( x_4 \) is 1 or 0, respectively, i.e.,

Product word\( = 16A + (\text{sign value}) \times (\text{APC word}) \tag{3}\)

Where sign value = 1 for \( x_4 = 1 \) and sign value = −1 for \( x_4 = 0 \). The product value for \( X = (10000) \) corresponds to APC value “zero,” which could be derived by resetting the LUT output, instead of storing that in the LUT.

**B. Modified OMS for LUT Optimization**

It is shown in [9] that, for the multiplication of any binary word X of size L, with a fixed coefficient A, instead of storing all the 2L possible values of \( C = AX \), only \( (2L/2) \) words corresponding to the odd multiples of A may be stored in the LUT, while all the even multiples of A could be derived by left-shift operations of one of those odd multiples. Based on the above assumptions, the LUT for the multiplication of an L-bit input with a W-bit coefficient could be designed by the following strategy.

1) A memory unit of \( [(2L/2) + 1] \) words of \( (W + L) \)-bit width is used to store the product values, where the first \( (2L/2) \) words are odd multiples of A, and the last word is zero.

2) A barrel shifter for producing a maximum of \( (L - 1) \) left shifts is used to derive all the even multiples of A.

3) The L-bit input word is mapped to the \( (L - 1) \)-bit address of the LUT by an address encoder, and control bits for the barrel shifter are derived by a control circuit.

As required by (3), the word to be stored for \( X = (00000) \) is not 0 but 16A, which we can obtain from A by four left shifts using a barrel shifter. However, if 16A is not derived from A, only a maximum of three left shifts is required to obtain all other even multiples of A. A maximum of three bit shifts can be implemented by a two-stage logarithmic barrel shifter, but the implementation of four shifts...
requires a three-stage barrel shifter. Therefore, it would be a more efficient strategy to store 2\(A\) for input \(X = (00000)\), so that the product 16\(A\) can be derived by three arithmetic left shifts.

The product values and encoded words for input words \(X = (00000)\) and \((10000)\) are separately shown in Table III. For \(X = (00000)\), the desired encoded word 16\(A\) is derived by 3-bit left shifts of 2\(A\) [stored at address \((10000)\)]. For \(X = (10000)\), the APC word “0” is derived by resetting the LUT output, by an active-high RESET signal given by

\[
\text{Reset} = (X_0 + X_1 + X_2 + X_3)X_4 \quad (4)
\]

It may be seen from Tables II and III that the 5-bit input word \(X\) can be mapped into a 4-bit LUT address \((d3d2d1d0)\), by a simple set of mapping relations

\[
d1 = xni + 1, \text{ for } i = 0,1,2, \ldots \text{ d3} = x^n \quad (5)
\]

where \(X'' = (x''3x''2x''1x''0)\) is generated by shifting-out all the leading zeros of \(X\) by an arithmetic right shift followed by address mapping, i.e.,

\[
X' = \begin{cases} 
Y_L, & \text{if } x4 = 1 \\
Y_L, & \text{if } x4 = 0
\end{cases} \quad (6)
\]

III. IMPLEMENTATION OF PROPOSED LUT OPTIMIZATION SCHEME USING MICROWIND

In this section, we discuss the implementation of the LUT-based multiplier using the proposed scheme, where the LUT is optimized by a combination of the proposed APC scheme and a modified OMS technique.

A. Implementation of the LUT Multiplier

Using APC for \(L = 5\). The structure and function of the LUT-based multiplier for \(L = 5\) using the APC technique is shown in Fig. 2. It consists of a four-input LUT of 16 words to store the APC values of product words as given in the sixth column of Table I, except on the last row, where 2\(A\) is stored for input \(X = (00000)\) instead of storing a “0” for input \(X = (10000)\). Besides, it consists of an address-mapping circuit and an add/subtract circuit. The address-mapping circuit generates the desired address \((x'3x'2x'1x'0)\) according to (2). A straightforward implementation of address mapping can be done by multiplexing \(XL\) and \(X' L\) using \(x4\) as the control bit.

The address-mapping circuit, however, can be optimized to be realized by three XOR gates, three AND gates, two OR gates, and a NOT gate, as shown in Fig. 2. Note that the RESET can be generated by a control circuit (not shown in this figure) according to (4). The output of the LUT is added with or subtracted from 16\(A\), for \(x4 = 1\) or 0, respectively, according to (3) by the add/subtract cell. Hence, \(x4\) is used as the control for the add/subtract cell.

B. Implementation of the Optimized LUT Using Microwind

The proposed APC–OMS combined design of the LUT for \(L = 5\) and for any coefficient width \(W\) is shown in Fig. 3. It consists of an LUT of nine words of \((W + 4)\)-bit width,
decoder, as shown in Fig. 4(a). The control bits $s_0$ and $s_1$ to be used by the barrel shifter to produce the desired number of shifts of the LUT output are generated by the control circuit, according to the relations

$$S_0 = x_0 + (x_1 + x_2) \quad (7a)$$
$$S_1 = (x_0 + x_1) \quad (7b)$$

Note that $(s_1s_0)$ is a 2-bit binary equivalent of the required number of shifts specified in Tables II and III.

The RESET signal given by (4) can alternatively be generated as $(d_3 \text{ AND } x_4)$. The control circuit to generate the control word and RESET is shown in Fig. 4(b).

![Fig. 4. (a) Four-to-nine-line address-decoder. (b)Control circuit for generation of $s_0$, $s_1$, and RESET.](image)

The address-generator circuit receives the 5-bit input operand $X$ and maps that onto the 4-bit address word $(d_3d_2d_1d_0)$. A simplified address generator is presented later in this section.

### C. Optimized LUT Design for Signed and Unsigned Operands

The APC–OMS combined optimization of the LUT can also be performed for signed values of $A$ and $X$. When both operands are in sign-magnitude form, the multiples of magnitude of the fixed coefficient are to be stored in the LUT, and the sign of the product could be obtained by the XOR operation of sign bits of both multiplicands. When both operands are in two’s complement forms, a two’s complement operation of the output of the LUT is required to be performed for $x_4 = 1$. There is no need to add the fixed value $16A$ in this case, because the product values are naturally in antisymmetric form.

Note that, except the last word, all other words in the LUT are odd multiples of $A$. The fixed coefficient could be even or odd, but if we assume $A$ to be an odd number, then the all the stored product words (except the last one) would be odd. If the stored value $P$ is an odd number, it can be expressed as

$$P = P_{D-1}P_{D-2}...P_11 \quad (8)$$

and its two’s complement is given by

$$P' = P'_{D-1}P'_{D-2}...P'_11 \quad (9)$$

where $P'_i$ is the one’s complement of $P_i$ for $1 \leq i \leq D - 1$, and $D = W + L - 1$ is the width of the stored words. If we store the two’s complement of all the product values and change the sign of the LUT output for $x_4 = 1$, then the sign of the last LUT word need not be changed. Based on (9), we can therefore have a simple sign-modification circuit [shown in Fig. 6(a)] when $A$ is an odd integer. However, the fixed coefficient $A$ could be even as well. When $A$ is a nonzero even integer, we can express it as $A' \times 2^l$, where $1 \leq l \leq D - 1$ is an integer, and $A'$ is an odd integer.

![Fig. 6. (a) Optimized implementation of the sign modification of the odd LUT output.](image)
Instead of storing multiples of \(A\), we can store multiples of \(A'\) in the LUT, and the LUT output can be left shifted by \(l\) bits by a hardwired shifter. Similarly, using (5) and (6), we can have an address-generation circuit as shown in Fig. 6(b), since all the shifted-address \(Y_L\) (except the last one) is an odd integer.

**IV. RESULTS AND DISCUSSION**

The proposed LUT multipliers for word size \(L = W = 8\), 16, and 32 bits are designed and executed using microwind and DSCH tools, where the LUTs are implemented as arrays of constants, and additions are implemented by the Wallace tree and ripple carry array.

The CSD-based multipliers having the same addition schemes are also synthesized with the same technology library. It is found that the proposed LUT design involves comparable area and time complexities for a word size of 8 bits, but for higher word sizes, it involves significantly less area and less multiplication time than the CSD-based multiplier. For \(L = W = 16\), and 32 bits, respectively, it offers more than 30\% and 50\% of saving in area–delay product (ADP) over the CSD multiplier.

In this brief, we have shown the possibility of using LUT based multipliers to implement the constant multiplication for DSP applications. The full advantages of proposed LUT based design, however, could be derived if the LUTs are implemented as NAND or NOR read-only memories and the arithmetic shifts are implemented by an array barrel shifter using metal–oxide–semiconductor transistors [11]. Further work could still be done to derive OMS–APC-based LUTs for higher input sizes with different forms of decompositions and parallel and pipelined addition schemes for suitable area–delay tradeoffs.

**REFERENCES**


**IX. BIOGRAPHY**

**U. Palani** was born in Tamilnadu on 1979 received his UG degree in Electronics Engineering from Madras University in 2001 and PG degree from
Vinayaga Mission University. His field of interest is Applications of Electrical and Electronics Engineering, Digital Electronics. He is having 10 years of teaching Experience. He is a Life member of ISTE.

M.Sujith was born in Namakkal, Tamilnadu in 1987. He received the B.E. degree in Electrical and Electronics Engineering from K.S.R. College of Engineering and received M.E. degree in Applied Electronics from the Annavi Mathamal Sheela Engineering College. He published 5 international journals and Presented papers in 6 national and international conferences. He is currently working at I.F.E.T College of Engineering as Senior Assistant Professor. His main research interests include power electronic converters, compensators and digital communications

P.Pugazhendiran was born in Tamilnadu, on 1979. Received his UG degree in Electrical and Electronics Engineering from Coimbatore Institute of Technology (CIT) in 2001 and PG degree from College of Engineering Guindy (CEG), Anna University, Chennai in 2009. His research interest includes Power quality issues, Power Converters, Renewable energy sources, Electrical Drives. He Published More than 5 Engineering Books. Teaching Experience over a decade. He published more than 10 National and International journals He is currently working at I.F.E.T College of Engineering as Associate Professor and head of the department. He is a life member of ISTE.