S.Jagadeesh, S.Venkata Chary** / International Journal Of Engineering Research And Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, issue 5, September-October 2012, pp.425-431

Design of Parallel Multiplier–Accumulator Based on Radix-4 Modified Booth Algorithm with SPST

S. JAGADEESH , S.VENKATA CHARY

1Associate Professor & HOD, Department of Electronics and Communication Engineering, Sri Sai Jyothi Engineering College, Gandipet, Hyderabad-75, (A. P.), India

2 P.G. Student, M.Tech. (VLSI), Department of Electronics and Communication Engineering, Sri Sai Jyothi Engineering College, Gandipet, Hyderabad-75, (A. P.), India

Abstract

In this paper, we proposed a new architecture of multiplier-and-accumulator (MAC) for high-speed arithmetic By combining multiplication with accumulation and devising a hybrid type of carry save adder (CSA), the performance was improved.

But using SPST(Spurious Power Suppression Technique) we can reduce power and the overall performance was elevated. The proposed SPST based radix-4 modified Booth's algorithm (MBA) and has the modified array for the sign extension in order to increase the bit density of the operands. The Booth's radix-4 algorithm, Modified Booth Multiplier improves speed of Multipliers and SPST adder will reduce the power consumption in addition process. Also, the proposed MAC accumulates the intermediate results in the type of sum and carrybits instead of the output of the final adder. Based on the theoretical and experimental estimation, we analyzed the results such as the amount of hardware resources, delay, and power. We expect that the proposed MAC can be adapted to various fields requiring high performance such as the signal processing areas.

Index Terms—Booth multiplier, carry save adder (CSA) tree, Spurious power suppression technique(SPST), Computer arithmetic, digital signal processing (DSP), multiplier and-Accumulator (MAC).

I. INTRODUCTION

With the recent rapid advances in multimedia and communication systems, real-time signal processing like audio signal processing, video/image processing, or large-capacity data processing are increasingly being demanded. The multiplier and multiplier-and-accumulator (MAC) [1] are the essential elements of the digital signal processing such as filtering, convolution, and inner products. Most digital signal processing methods use nonlinear functions such as discrete cosine transform (DCT) [2] or discrete wavelet transform (DWT)[3]. Because they are basically accomplished by repetitive application of multiplication and addition, the speed of the multiplication and addition arithmetic's determines the execution speed and performance of the entire calculation. Because the multiplier requires the longest delay among the basic operational blocks in digital system, the critical path is determined by the multiplier, in general. For high-speed multiplication, the modified radix-4 Booth's algorithm (MBA) [4] is commonly used. However, this cannot completely solve the problem due to the long critical path for multiplication [5], [6].

In general, a multiplier uses Booth's algorithm [7] and array of full adders (FAs), or Wallace tree [8] instead of the array of FAs. i.e., this multiplier mainly consists of the three parts:

Booth encoder, a tree to compress the partial products such as based on Radix-4 modified booth multiplier and final adder [9], [10]. We can use SPST technique to reduce power consumption. The most effective way to increase the speed of a multiplier is to reduce the number of the partial products by Radix-4 modified booth multiplier. Because multiplication proceeds a series of additions for the partial products. To reduce the number of calculation steps for the partial products, MBA algorithm has been applied. To increase the speed of the MBA algorithm, many parallel multiplication architectures have been researched [11]–[13].

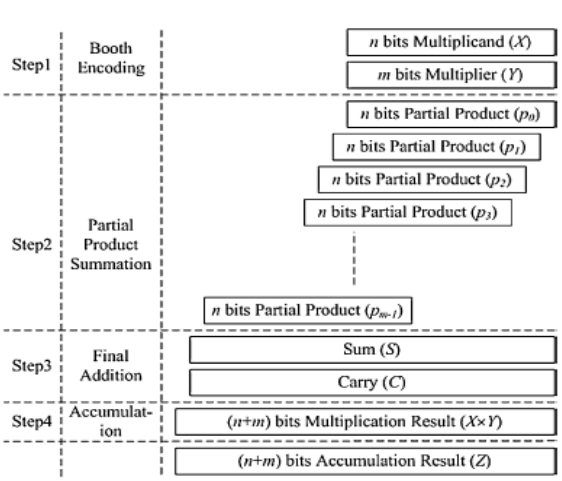


Fig1. Basic arithmetic steps of multiplication and accumulation

One of the most advanced types of MAC for general-purpose digital signal processing has been proposed by Elguibaly [14].While it has a better performance because of the reduced critical path compared to the previous MAC architectures, there is a need to improve the output rate due to the use of the final adder results for accumulation. An architecture to merge the adder block to the accumulator register in the MAC operator was proposed in [15] to provide the possibility of using two separate N/2-bit adders instead of one -bit adder to accumulate the N–bit MAC results. Recently, Zicari proposed an architecture that took a merging technique to fully utilize the 4–2 compressor [16].It also took this compressor as the basic building blocks for the multiplication circuit.

In this paper, a new architecture for a high-speed MAC is proposed. In this MAC, the computations of multiplication and accumulation are combined and a hybrid-type SPST structure is proposed to reduce the critical path and power improve the output rate. It uses MBA algorithm based on 1's complement number system. A modified array structure for the sign bits is used to increase the density of the operands. In addition, in order to increase the output rate by optimizing the pipeline efficiency, intermediate calculation results are accumulated in the form of sum and carry instead of the final adder outputs.

This paper is organized as follows. In Section II, a simple introduction of a general MAC will be given, and the architecture for the proposed MAC will be described in Section III. In Section IV, the proposed SPST will be analyzed and the characteristic of the proposed MAC will be shown. Finally, the conclusion will be given in Section V.

II. OVERVIEW OF MAC

In this section, basic MAC operation is introduced. A multiplier can be divided into three operational steps. The first is radix-4 Booth encoding in which a partial product is generated from the multiplicand and the multiplier. The second is adder array or partial product compression to add all partial products and convert them into the form of sum and carry. The last is the final addition in which the final multiplication result is produced by adding the sum and the carry. If the process to accumulate the multiplied results is included, a MAC consists of four steps, as shown in Fig. 1, which shows the operational steps explicitly.

A general hardware architecture of this MAC is shown in Fig 2. It executes the multiplication operation by multiplying the input multiplier and the multiplicand . This is added to the previous multiplication result as the accumulation step.

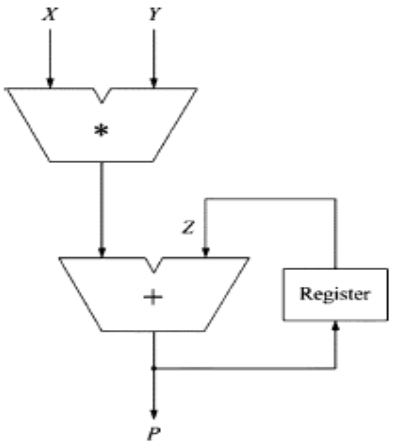


Fig.2. Hardware architecture of general MAC.

The N-bit 2's complement binary number can be expressed as

$$X = -2^{N-1}x_{N-1} + \sum_{i=0}^{N-2} x_i 2^i, \qquad x_i \in 0,1. \quad (1)$$

If (1) is expressed in base-4 type redundant sign digit form in order to apply the radix-2 Booth's algorithm, it would be [7].

$$X = \sum_{i=0}^{N/2-1} d_i 4_i \quad \ldots\ldots (2)$$

$$d_i = -2x_{2i+1} + x_{2i} + x_{2i-1}. \quad \ldots (3)$$

If (2) is used, multiplication can be expressed as

$$X \times Y = \sum_{i=0}^{N/2-1} d_i 2^{2i} Y. \quad \ldots\ldots\ldots (4)$$

If these equations are used, the afore-mentioned multiplication–accumulation results can be expressed as

$$P = X \times Y + Z = \sum_{i=0}^{N/2-1} d_i 2^i Y + \sum_{j=0}^{2N-1} z_i 2^i. \quad (5)$$

Each of the two terms on the right-hand side of (5) is calculated independently and the final result is produced by adding the two results. The MAC architecture implemented by (5) is called the standard design [6]. If N-bit data are multiplied, the number of the generated partial products is proportional to N. In order to add them serially, the execution time is also proportional to N. The architecture of a multiplier, which is the fastest, uses radix-4 both encoding that generates partial products and a Wallace tree based on CSA as the adder array to add the partial products. If radix-4 Booth encoding is used, the number of partial products, i.e., the inputs to the Wallace tree, is reduced to half,

resulting in the decrease in CSA tree step. In addition, the signed multiplication based on 2's complement numbers is also possible. Due to these reasons, most current used multipliers adopt the Booth encoding

III. PROPOSED MAC ARCHITECTURE

In the majority of digital signal processing (DSP) applications the critical operations usually involve many multiplications and/or accumulations. For real-time signal processing, a high speed and high throughput Multiplier-Accumulator (MAC) is always a key to achieve a high performance digital signal processing system. In the last few years, the main consideration of MAC design is to enhance its speed. This is because; speed and throughput rate is always the concern of digital signal processing system. But for the epoch of personal communication, low power design also becomes another main design consideration. This is because; battery energy available for these portable products limits the power consumption of the system. Therefore, the main motivation of this work is to investigate various Pipelined multiplier/accumulator architectures and circuit design techniques which are suitable for implementing high throughput signal processing algorithms and at the same time achieve low power consumption. A conventional MAC unit consists of (fast multiplier) multiplier and an accumulator that contains the sum of the previous consecutive products. The function of the MAC unit is given by the following equation:

$$F= \Sigma AiBi \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

The main goal of a DSP processor design is to enhance the speed of the MAC unit, and at the same time limit the power consumption. In a pipelined MAC circuit, the delay of pipeline stage is the delay of a 1-bit full adder. Estimating this delay will assist in identifying the overall delay of the pipelined MAC. In this work, 1-bit full adder is designed. Area, power and delay are calculated for the full adder, MAC unit is designed for low power.

High-Speed Booth Encoded Parallel Multiplier Design

Fast multipliers are essential parts of digital signal processing systems. The speed of multiply operation is of great importance in digital signal processing as well as in the general purpose processors today, especially since the media processing took off. In the past multiplication was generally implemented via a sequence of addition, subtraction, and shift operations. Multiplication can be considered as a series of repeated additions. The number to be added is the multiplicand, the number of times that it is added is the multiplier, and the result is the product. Each step of addition generates a partial product. In most computers, the operand usually contains the same number of bits. When the operands are interpreted as integers, the product is generally twice the length of operands in order to preserve the information content. This repeated addition method that is suggested by the arithmetic definition is slow that it is almost always replaced by an algorithm that makes use of positional representation. It is possible to decompose multipliers into two parts. The first part is dedicated to the generation of partial products, and the second one collects and adds them. The basic multiplication principle is two fold i.e. evaluation of partial products and accumulation of the shifted partial products. It is performed by the successive additions of the columns of the shifted partial product matrix. The 'multiplier' is successfully shifted and gates the appropriate bit of the 'multiplicand'. The delayed, gated instance of the multiplicand must all be in the same column of the shifted partial product matrix. They are then added to form the product bit for the particular form. Multiplication is therefore a multi operand operation. To extend the multiplication to both signed and unsigned.

Modified Booth Encoder:

In order to achieve high-speed multiplication, multiplication algorithms using parallel counters, such as the modified Booth algorithm has been proposed, and some multipliers based on the algorithms have been implemented for practical use. This type of multiplier operates much faster than an array multiplier for longer operands because its computation time is proportional to the logarithm of the word length of operands.

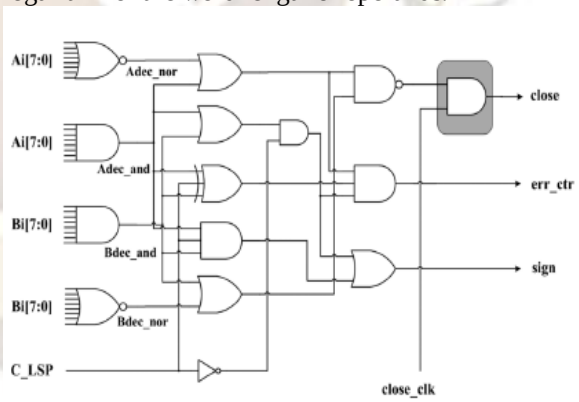


Fig 3. Modified Booth Encoder

Booth multiplication is a technique that allows for smaller, faster multiplication circuits, by recoding the numbers that are multiplied. It is possible to reduce the number of partial products by half, by using the technique of radix-4 Booth recoding. The basic idea is that, instead of shifting and adding for every column of the multiplier term

and multiplying by 1 or 0, we only take every second column, and multiply by ±1, ±2, or 0, to obtain the same results. The advantage of this method is the halving of the number of partial products. To Booth recode the multiplier term, we consider the bits in blocks of three, such that each block overlaps the previous block by one bit. Grouping starts from the LSB, and the first block only uses two bits of the multiplier. Figure 3 shows the grouping of bits from the multiplier term for use in modified booth encoding.

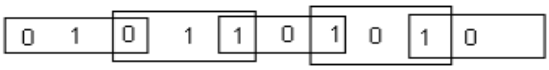


Fig.2.2 Grouping of bits from the multiplier term

Each block is decoded to generate the correct partial product. The encoding of the multiplier Y, using the modified booth algorithm, generates the following five signed digits, -2, -1, 0, +1, +2. Each encoded digit in the multiplier performs a certain operation on the multiplicand, X, as illustrated in Table 1

Block	Re - coded digit	Operation on X
000	0	0 X
001	+1	+1 X
010	+1	+1 X
011	+2	+2 X
100	-2	-2 X
101	-1	-1 X
110	-1	-1 X
111	0	0 X

For the partial product generation, we adopt Radix-4 Modified Booth algorithm to reduce the number of partial products for roughly one half. For multiplication of 2's complement numbers, the two-bit encoding using this algorithm scans a triplet of bits. When the multiplier B is divided into groups of two bits, the algorithm is applied to this group of divided bits. Figure 4, shows a computing example of Booth multiplying two numbers "2AC9" and "006A". The shadow denotes that the numbers in this part of Booth multiplication are all zero so that this part of the computations can be neglected. Saving those computations can significantly reduce the power consumption caused by the transient signals. According to the analysis of the multiplication shown in figure 4, we propose the SPST-equipped modified-Booth encoder, which is controlled by a detection unit. The detection unit has one of the two operands as its input to decide whether the Booth encoder calculates redundant computations. As shown in figure 9. The latches can, respectively, freeze the inputs of MUX-4 to MUX-7 or only those of MUX-6 to MUX-7 when the PP4 to PP7 or the PP6 to PP7 are zero; to reduce the transition power dissipation. Figure 10, shows the booth partial product generation circuit. It includes AND/OR/EX-OR logic.

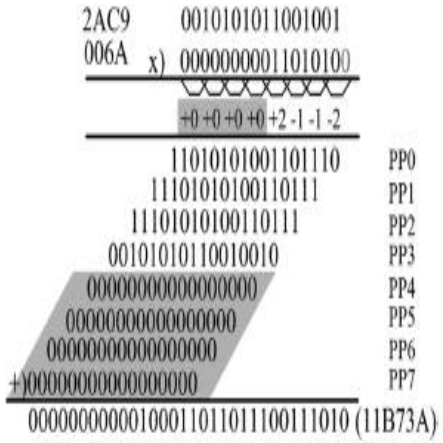


Fig.4 Illustration of multiplication using modified Booth encoding

II. PROPOSED SPST

Besides the explanations presented in our former studies this paper provides further illustrations of the proposed SPST as described in the following sections.

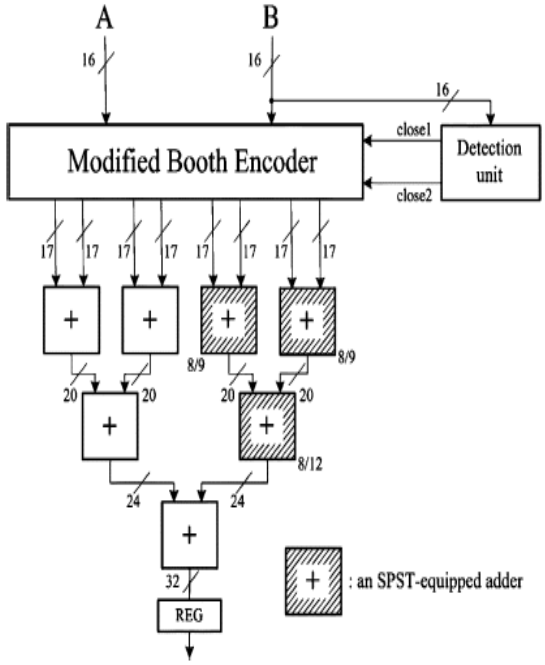


Fig4: Architecture for Multiplication with SPST adder

A. Theoretical Analysis and Logic Derivation

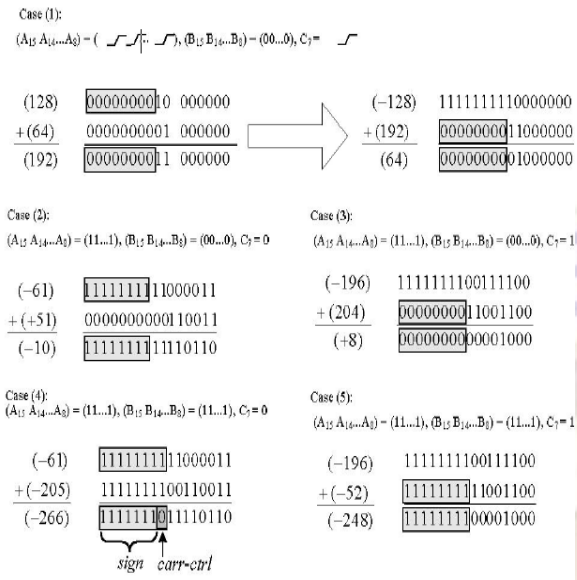


Fig 5:Spurious transitions in the multimedia/DSP computations.

Hence, there is probably no other case beyond these five based on this design. The first case illustrates a transient state in which spurious transitions of carry signals occur in the MSP, although the final result of the MSP is unchanged. Meanwhile, the second and third cases describe situations involving one negative operand adding another positive operand without and with carry-in from the LSP, respectively. Moreover, the fourth and fifth cases demonstrate the addition of two negative operands without and with carry-in from the LSP, respectively. In those cases, the results of MSP are predictable; therefore, the computations in MSP are useless and can be neglected. Eliminating those spurious computations not only can save the power consumption inside the adder/subtractor in the current stage but also can decrease the glitching noises which cause power wastage inside the arithmetic circuits in the next stage. From the analysis of Fig. 5, we are motivated to propose the SPST that separates the adder/subtractor into two parts and then latches the input data of the MSP whenever they do not affect the computation results.

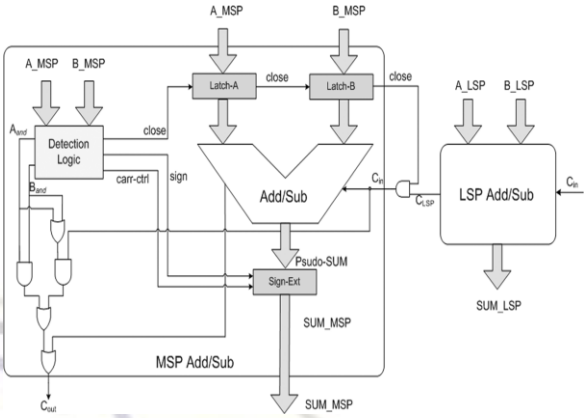


Fig 6: A 16-bitadder/subtractor design example adopting the proposed SPST.

The SPST can be expanded to be a fine-grain scheme in which the adder/subtractor is divided into more than two parts. However the hardware complexity of the augmented circuits such as the detection-logic unit, the data latches, and the SE unit increases dramatically. Based on an adder/subtractor example, we actually find that the power expense caused by the augmented circuits is larger than the power reduction in a tripartitioned scheme. This is the reason. we propose a bipartitioned SPST scheme in this paper. To know whether the MSP affects the computation results in the bipartitioned SPST scheme, a detection-logic unit must be used to detect the effective input ranges.

$$\begin{aligned}\text{Carr\_ctrl} = {} & \overline{C_{LSP}} \cdot A_{and} \cdot A_{nor} \cdot B_{and} \cdot \overline{B_{nor}} + \overline{C_{LSP}} \cdot \\ & A_{and} \cdot \overline{A_{nor}} \cdot \overline{B_{and}} \cdot B_{nor} + C_{LSP} \cdot \\ & \overline{A_{and}} \cdot A_{nor} \cdot \overline{B_{and}} \cdot B_{nor} + C_{LSP} \cdot A_{and} \cdot \\ & \overline{A_{nor}} \cdot B_{and} \cdot \overline{B_{nor}}; \\ = {} & C_{LSP} (\overline{A_{and}} \cdot B_{and} + A_{and} \cdot \overline{B_{and}}) \cdot (A_{and} \cdot B_{and} \\ & + A_{and} \cdot B_{nor} + A_{nor} \cdot B_{and} + A_{nor} \cdot B_{nor}) + \\ & C_{LSP} \cdot (A_{and} \cdot B_{and} + A_{and} \cdot B_{and} + A_{and} \cdot B_{nor} + \\ & A_{nor} \cdot B_{and} + A_{nor}) \\ = {} & (C_{LSP} \oplus A_{and} \oplus B_{and}) \cdot (A_{and} + A_{nor}) \cdot \\ & (B_{and} + B_{nor}); \\ \text{sign} = {} & \overline{C_{LSP}} \cdot (A_{and} + B_{and}) + C_{LSP} \cdot A_{and} \cdot B_{and};\end{aligned}$$

Fig.7Representations of *carry-ctrl* signal and *sign* signal in terms of *KARNAUGH* maps logical extensions.

where A[m] and B[n] , respectively, denote the m th bit of the operand A and the n th bit of the operand B, and AMSP and BMSP,respectively, denote the MSP parts, i.e., the 9th bit to the 16th bit,

of the operands A and B in the examples shown in Fig. 5. When the bits in AMSP and/or BMSP in are all ones, the value of Aand and/or that of Band, respectively, become one, while when the bits in AMS and/or BMSP in are all zeros, the value of Anor and/or that of Bnor, respectively, turn into one. Being one of the three outputs of the detection-logic unit, *close* denotes whether the MSP circuits can be neglected or not. When the two input operands can be classified into one of the five cases shown in Fig. 5, the value of *close* becomes zero, which indicates that the MSP circuits can be closed to save power dissipation.

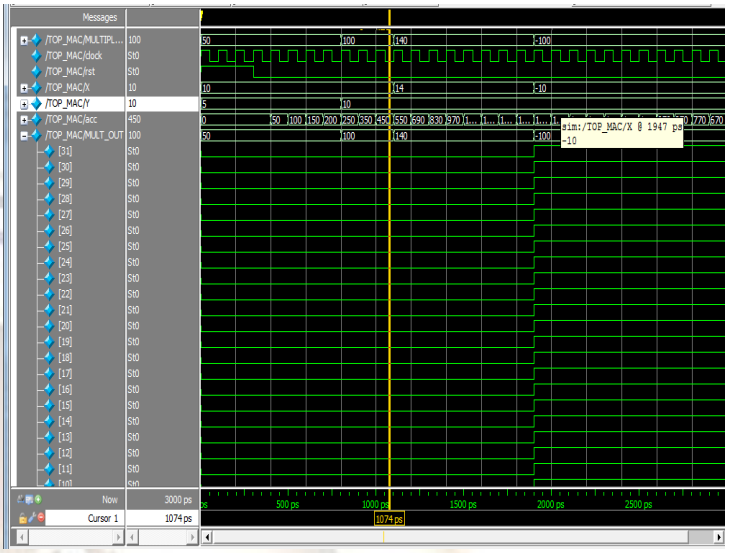
This design intends to close the MSP circuits by feeding zero inputs into them, which may freeze the switching activities in the MSP circuits to avoid dynamic power consumption. Compared with the use of transmission gates to latch the inputs, this scheme can prevent the voltage-drop problems caused by the floating-connected points after the MSP circuits are closed for a relatively long span of time. The ways to compensate for the sign bits of the computing results are also shown in case 4 in Fig. 3. Accordingly, we derive the *KARNAUGH* maps shown in Fig. 6 which lead to the Boolean logical equations

V. CONCLUSION

In this paper, a new MAC architecture to execute the multiplication - accumulation operation, which is the key operation, for digital signal processing and multimedia information processing efficiently, was proposed. By removing the independent accumulation process that has the largest delay and merging it to the compression process of the partial products, the overall MAC performance has been improved almost twice as much as in the previous work.

The proposed MAC required the hardware resources as much as the previous research. While the delay has been increased slightly compared to the previous research, actual power has been reduced by using SPST adder technique in addition of our partial products of MAC operation. This paper also proposes a low-power technique called SPST and explores its applications in multimedia/DSP computations, where the theoretical analysis and the realization issues of the SPST are fully discussed. The proposed SPST can obviously decrease the switching (or dynamic) power dissipation, which comprises a significant portion of the whole power dissipation in integrated circuits. the proposed SPST can save 27% power consumption at the cost of only 20% area overheads. Besides, the proposed SPST can achieve a 24% saving in power consumption at the expense of only 10% area. Consequently, we can expect that the proposed architecture can be used effectively in the area requiring high throughput such as a real-time digital signal processing.

VI. RESULTS




```
Power summary:                   I(mA)    P(mW)
---------------------------------------------------------------
Total estimated power consumption:              203
                                 ---
                      Vccint 1.20V:      69      83
                      Vccaux 2.50V:      45     113
                      Vcco25 2.50V:       3       8
                                 ---
                           Clocks:       0       0
                           Inputs:       0       0
                            Logic:       0       0
                          Outputs:
                            Vcco25       0       0
                          Signals:       0       0
                                 ---
          Quiescent Vccint  1.20V:      69      83
          Quiescent Vccaux  2.50V:      45     113
          Quiescent Vcco25  2.50V:       3       8

Thermal summary:
---------------------------------------------------------------
   Estimated junction temperature:              29C
                  Ambient temp:  25C
                     Case temp:  28C
                     Theta J-A:  19C/W

Analysis completed: Sun Sep 16 16:38:38 2012
---------------------------------------------------------------
```

Design Summary | Synthesis Report | TOP_MAC.ngr | Map Report | Place and Route Repor | Clock Region Repor | Power Report

REFERENCES

[1] J. J. F. Cavanagh, *Digital Computer Arithmetic. New York: McGraw-Hill, 1984.*

[2] Information Technology-Coding of Moving Picture and Associated Autio, MPEG-2 Draft International Standard, ISO/IEC 13818-1, 2, 3, 1994.

[3] *JPEG 2000 Part I Fina1119l Draft,* ISO/IEC JTC1/SC29 WG1.

[4] O. L. MacSorley, "High speed arithmetic in binary computers," *Proc.IRE*, vol. 49, pp. 67–91, Jan. 1961.

[5] S. Waser and M. J. Flynn, *Introduction to Arithmetic for Digital Systems Designers*. New York: Holt, Rinehart and Winston, 1982.

[6] A. R. Omondi, *Computer Arithmetic Systems*. Englewood Cliffs, NJ:Prentice-Hall, 1994.

[7] A. D. Booth, "A signed binary multiplication technique," *Quart. J.Math.*, vol. IV, pp. 236–240, 1952.

[8] C. S. Wallace, "A suggestion for a fast multiplier," *IEEE Trans. Electron Computer.*, vol. EC-13, no. 1, pp. 14–17, Feb. 1964.

[9] A. R. Cooper, "Parallel architecture modified Booth multiplier," *Proc.Inst. Electr. Eng. G*, vol. 135, pp. 125–128, 1988.

[10] N. R. Shanbag and P. Juneja, "Parallel implementation of a 4□4-bit multiplier using modified Booth's algorithm," *IEEE J. Solid-State Circuits*, vol. 23, no. 4, pp. 1010–1013, Aug. 1988.

[11] G. Goto, T. Sato, M. Nakajima, and T. Sukemura, "A 54□54 regular structured tree multiplier," *IEEE J. Solid-State Circuits*, vol. 27, no. 9, pp. 1229–1236, Sep. 1992.

[12] J. Fadavi-Ardekani, "M□N Booth encoded multiplier generator using optimizedWallace trees," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 1, no. 2, pp. 120–125, Jun. 1993.

[13] N. Ohkubo, M. Suzuki, T. Shinbo, T. Yamanaka, A. Shimizu, K.Sasaki, and Y. Nakagome, "A 4.4 ns CMOS 54□54 multiplier using pass-transistor multiplexer," *IEEE J. Solid-State Circuits*, vol. 30, no. 3, pp. 251–257, Mar. 1995.

[14] F. Elguibaly, "A fast parallel multiplier–accumulator using the modified Booth algorithm," *IEEE Trans. Circuits Syst.*, vol. 27, no. 9, pp. 902–908, Sep. 2000.

[15] A. Fayed and M. Bayoumi, "A merged multiplier-accumulator for high speed signal processing applications," *Proc. ICASSP*, vol. 3, pp. 3212–3215, 2002.

[16] P. Zicari, S. Perri, P. Corsonello, and G. Cocorullo, "An optimized adder accumulator for high speed MACs," *Proc. ASICON 2005*, vol. 2, pp. 757–760, 2005.

[22] Z. Huang and M. D. Ercegovac, "High-performance low-power left-toright array multiplier design," *IEEE Trans. Comput.*, vol. 54, no. 3, pp. 272–283, Mar. 2005.

[23] M. C.Wen, S. J. Wang, and Y. N. Lin, "Low-power parallel multiplier with column bypassing," *Electron. Lett.*, vol. 41, no. 12, pp. 581–583, May 2005.

[24] H. Lee, "A power-aware scalable pipelined Booth multiplier," in *Proc.IEEE Int. SOC Conf.*, Sep. 2004, pp. 123–126.

[25] Y. Liao and D. B. Roberts, "A high-performance and low-power 32-bit multiply-accumulate unit with single-instruction–multiple-data (SIMD) feature," *IEEE J. Solid-State Circuits*, vol. 37, no. 7, pp. 926–931, Jul. 2002.

[26] A. Danysh and D. Tan, "Architecture and implementation of a vector/ SIMD multiply-accumulate unit," *IEEE Trans. Comput.*, vol. 54, no. 3, pp. 284–293, Mar. 2005.

[27] J. S.Wang, C. N. Kuo, and T. H. Yang, "Low-power fixed-width array multipliers," in *Proc. IEEE Symp. Low Power Electron. Des.*, Aug.9–11, 2004, pp. 307–312.

[28] W. C. Yeh and C. W. Jen, "High-speed Booth encoded parallel multiplier design," *IEEE Trans. Comput.*, vol. 49, no. 7, pp. 692–701, Jul.2000.