International Journal of Modern Engineering Research (IJMER)
www.ijmer.com Vol. 2, Issue. 5, Sep.-Oct. 2012 pp-3390-3397 ISSN: 2249-6645

Implementation of Low Power and High Speed Multiplier-Accumulator Using SPST Adder and Verilog

H. S. Krishnaprasad Puttam[1], P. Sivadurga Rao[2] & N. V. G. Prasad[3]
1, 2,3Department Of E.C.E,Sasi Institute Of Technology & Engineering,Tadepalligudem.India.
1M.Tech Student, 2Assistant Professor & 3Associate Professor,H.O.D.

Abstract: *In this paper, we proposed a new architecture of multiplier -and- accumulator (MAC) for high-speed arithmetic and low power. Multiplication occurs frequently in finite impulse response filters, fast Fourier transforms, discrete cosine transforms, convolution, and other important DSP and multimedia kernels. The objective of a good multiplier and accumulator (MAC) is to provide a physically compact, good speed and low power consuming chip. To save significant power consumption of a VLSI design, it is a good direction to reduce its dynamic power that is the major part of total power dissipation. In this paper, we propose a high speed MAC adopting the new SPST implementing approach. This multiplier and accumulator is designed by equipping the Spurious Power Suppression Technique (SPST) on a modified Booth encoder which is controlled by a detection unit using an AND gate. The modified booth encoder will reduce the number of partial products generated by a factor of 2. The SPST adder will avoid the unwanted addition and thus minimize the switching power dissipation.*

Keywords: *Booth encoder, computer arithmetic, digital signal processing, spurious power suppression technique, low power.*

I. INTRODUCTION

With the recent rapid advances in multimedia and communication systems, real-time signal processing's 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].

Power dissipation is recognized as a critical parameter in modern VLSI design field. To satisfy MOORE'S law and to produce consumer electronics goods with more backup and less weight, low power VLSI design is necessary.

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 twofold 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 numbers, a convenient number system would be the representation of numbers in two's complement format.

II. OVER VIEW OF MAC

In this section, basic MAC operation is introduced. A multiplier can be divided into three operational steps. The first is radix-2 Booth encoding in which a partial product is generated from the multiplicand (X) and the multiplier (Y). 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.

General hardware architecture of this MAC is shown in Fig. 2. It executes the multiplication operation by multiplying the input multiplier (X) and the multiplicand (Y) . This is added to the previous multiplication result (Z) as the accumulation step.

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

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

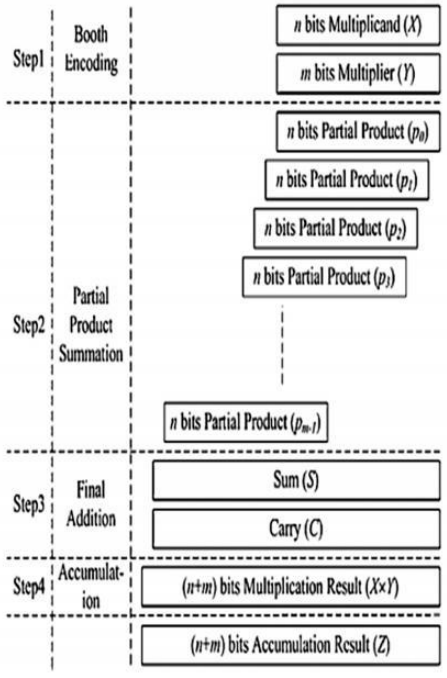


Figure 1: Basic Arithmetic Steps of Multiplication and Accumulation

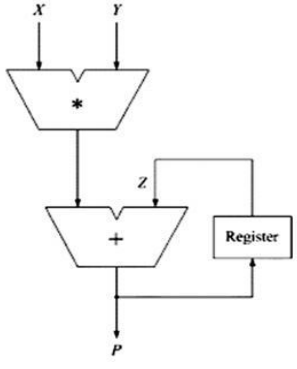


Figure 2. Hardware Architecture of General MAC

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].

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

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

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

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

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

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

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. 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-2 Booth encoding that generates partial products.

III. MODIFIED BOOTH ENCODER

In order to achieve high-speed, 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.

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 [8]. 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.

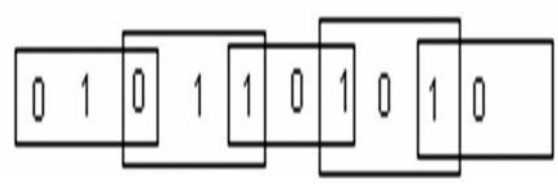


Figure 3. Grouping of Bits from the Multiplier Term

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.

Table 1
Encoded Five Signed Digits

Block	*Re-code digit*	*Operation on X*
001	0	0X
001	+1	+1X
010	+1	+1X
011	+2	+2X
100	–2	–2X
101	–1	–1X
110	–1	–1X
111	0	0X

IV. SPURIOUS POWER SUPPRESSION TECHNIQUE

The former SPST has been discussed in [9] and [10]. Figure 4 shows the five cases of a 16-bit addition in which the spurious switching activities occur. The 1st case illustrates a transient state in which the spurious transitions of carry signals occur in the MSP though the final result of the MSP are unchanged. The 2nd and the 3rd cases describe the situations of one negative operand adding another positive operand without and with carry from LSP, respectively. Moreover, the 4th and the 5th cases respectively demonstrate the addition of two negative operands without and with carry-in from LSP. In those cases, the results of the MSP are predictable Therefore the computations in the MSP are useless and can be neglected. The data are separated into the Most Significant Part (MSP) and the Least Significant Part (LSP). To know whether the MSP affects the computation results or not. We need a detection logic unit to detect the effective ranges of the inputs.

$$^{A}\text{MSP} = A\,[15:8] \quad (6)$$

$$^{B}\text{MSP} = B\,[15:8] \quad (7)$$

$$^{A}\text{and} = A\,[15].\,A\,[14] \ldots A\,[8] \quad (8)$$

$$^{B}\text{and} = B\,[15]\,.\,B\,[14] \ldots B\,[8] \quad (9)$$

$$^{A}\text{nor} = \overline{A[15] + A[14] + A[13] + \ldots\ldots + A[8]} \quad (10)$$

$$^{B}\text{nor} = \overline{B[15] + B[14] + B[13] + \ldots\ldots + B[8]} \quad (11)$$

$$\text{Close} = \sim ((A_{\text{and}} + A_{\text{nor}}).\,(B_{\text{and}} + B_{\text{nor}})) \quad (12)$$

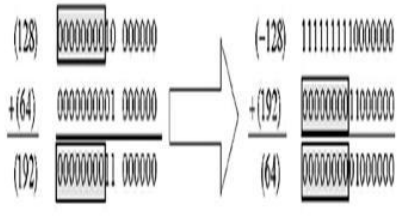
We derive the Karnaugh maps which lead to the Boolean equations (13) and (14) for the Carr_ctrl and the sign signals, respectively. In equation (13) and (14), C_{LSP} denotes the carry propagated from the LSP circuits.

$$\text{Carr\_ctrl} = (C_{LSP} \oplus A_{\text{and}} \oplus B_{\text{and}})\,(A_{\text{and}} + A_{\text{nor}})\,(B_{\text{and}} + B_{\text{nor}}) \quad (13)$$

$$\text{Sign} = C_{LSP}.\text{bar}\,(A_{\text{and}} + B_{\text{and}})$$

$$C_{LSP}\; A_{\text{and}}\; B_{\text{and}} \quad (14)$$

Case 1:



Case 2:

$$(A_{15}\,A_{14}\ldots A_8) = (11\ldots1),\ (B_{15}\,B_{14}\ldots B_8) = (00\ldots0),\ C_7 = 0$$

$$\begin{array}{ll} (-61) & 11111111\,11000011 \\ +(+51) & 00000000\,00110011 \\ \hline (-10) & 11111111\,11110110 \end{array}$$

Case 3:

$(A_{15}\ A_{14}...A_8) = (11...1), (B_{15}\ B_{14}...B_8) = (00...0), C_?= 1$

(−196)	1111111100111100
+(204)	0000000011001100
(+8)	0000000000001000

Case 4:

$(A_{15}\ A_{14}...A_8) = (11...1), (B_{15}\ B_{14}...B_8) = (11...1), C_?= 0$

(−61)	1111111110000011
+(−205)	1111111100110011
(−266)	1111111011110110

sign *carr-ctrl*

Case 5:

$(A_{15}\ A_{14}...A_8) = (11...1), (B_{15}\ B_{14}...B_8) = (11...1), C_?= 1$

(−196)	1111111100111100
+(−52)	1111111111001100
(−248)	1111111100001000

Figure 4: Spurious Transition Cases in Multimedia/DSP Processing

V. PROPOSED SPURIOUS POWER SUPPRESSION TECHNIQUE

The SPST uses a detection logic circuit to detect the effective data range of arithmetic units, e.g., adders or multipliers. When a portion of data does not affect the final computing results, the data controlling circuits of the SPST latch this portion to avoid useless data transitions occurring inside the arithmetic units. Besides, there is a data asserting control realized by using registers to further filter out the useless spurious signals of arithmetic unit every time when the latched portion is being turned on. This asserting control brings evident power reduction. Figure 5 shows the design of low power adder/subtractor with SPST.

The adder /subtractor is divided into two parts, the most significant part (MSP) and the least significant part (LSP). The MSP of the original adder/subtractor is modified to include detection logic circuits, data controlling circuits, sign extension circuits, logics for calculating carry in and carry out signals. The most important part of this study is the design of the control signal asserting circuits, denoted as asserting circuits in Figure 5. Although this asserting circuit brings evident power reduction, it may induce additional delay. There are two implementing approaches for the control signal assertion circuits. The first implementing approach of control signal assertion circuit is using registers. This is illustrated in Figure 6. The three output signals of the detection logic are close, Carr_ctrl, sign. The three output signals the detection logic unit are given a certain amount of delay before they assert. The delay □ , used to assert the three output signals, must be set in a range of □ <□ <□ , where□ denotes the data transient period and ? denotes the earliest required time of all the inputs. This will filter out the glitch signals as well as to keep the computation results correct. The restriction that □ □ must be greater than □ to guarantee the registers from latching the wrong values of control usually decreases the overall speed of the applied designs.

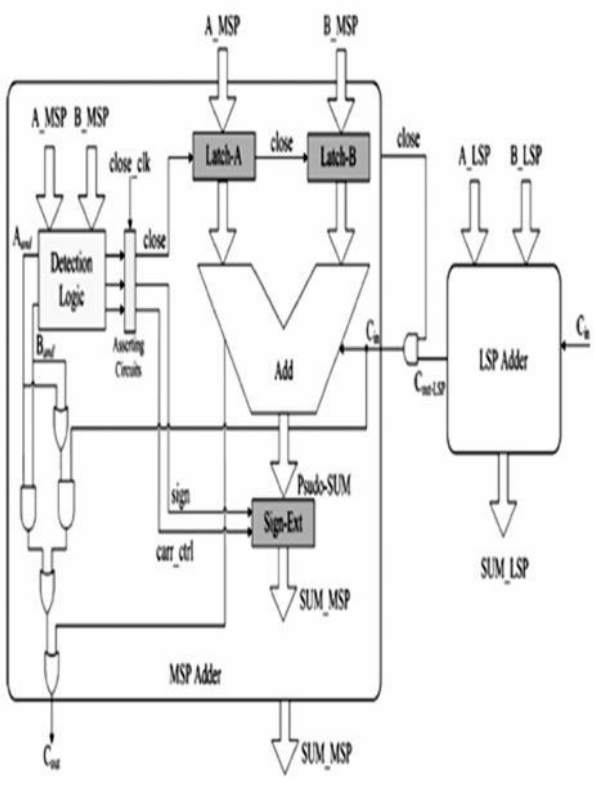


Figure 5. Low Power Adder/Subtractor Adopting the SPST

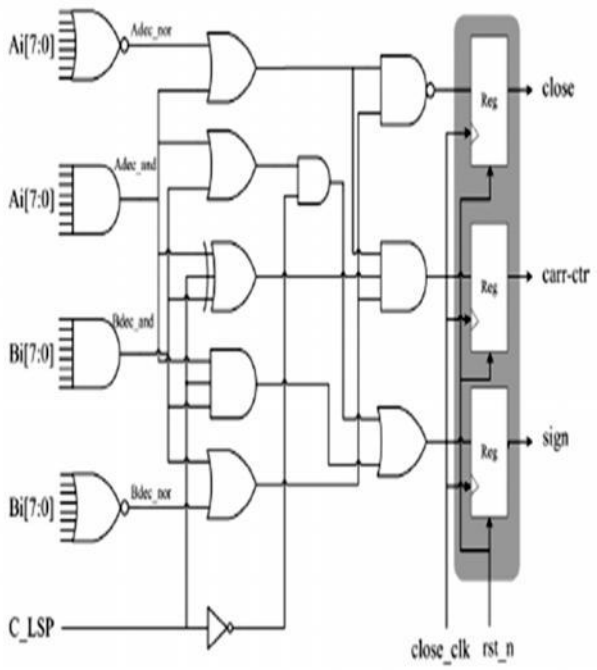


Figure 6. Detection Logic Circuits using Registers

This issue should be noticed in high- end applications which demands both high speed and low power requirements. To solve this problem we adopt the other implementing approach of control signal assertion circuit using AND gate.

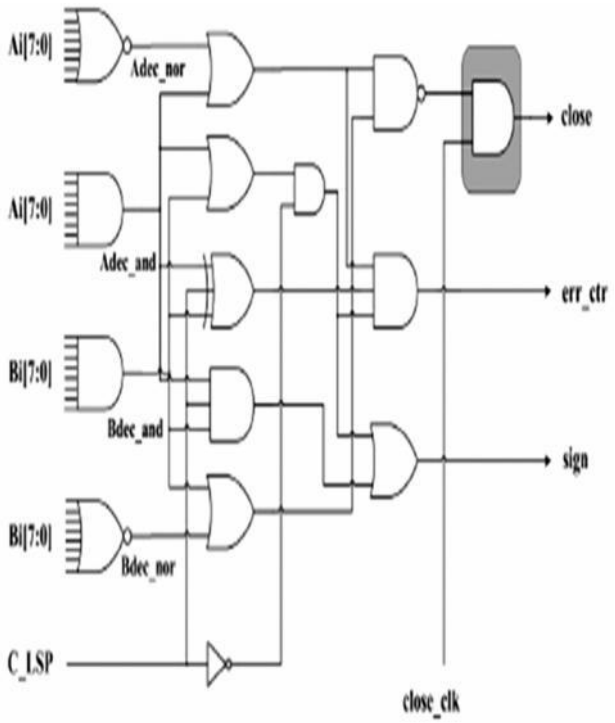


Figure 7. Detection Logic Circuits using AND Gate

VI. PROPOSED LOW POWER HIGH PERFORMANCE MULTIPLIER AND ACCUMULATOR

The proposed high speed low power multiplier is designed by equipping the SPST on a tree multiplier. There are two distinguishing design considerations in designing the proposed multiplier as listed in the following

(A) Applying the SPST on the Modified Booth Encoder

Figure 8 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 8, 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.

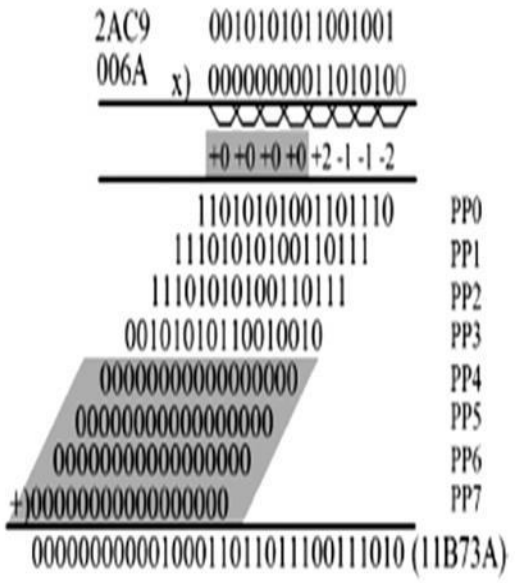


Figure 8: Illustration of Multiplication using Modified Booth Encoding

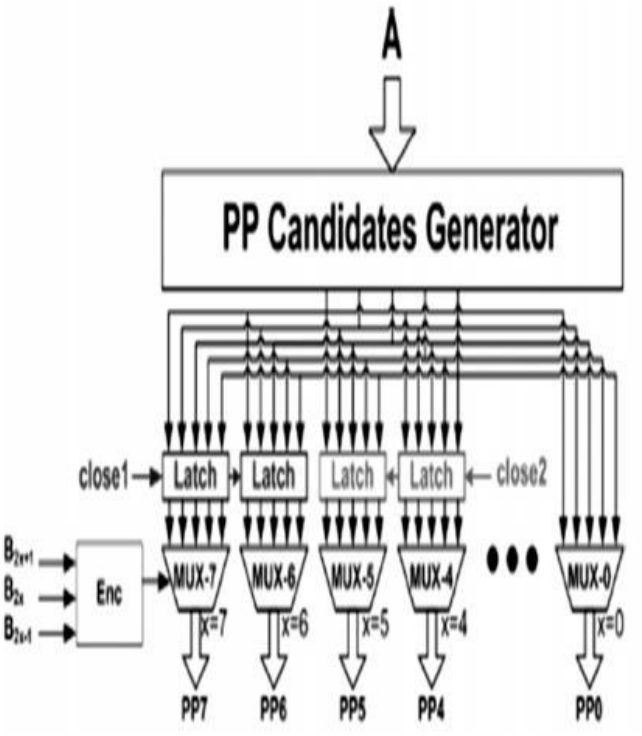


Figure 9: SPST Equipped Modified BoothEncoder

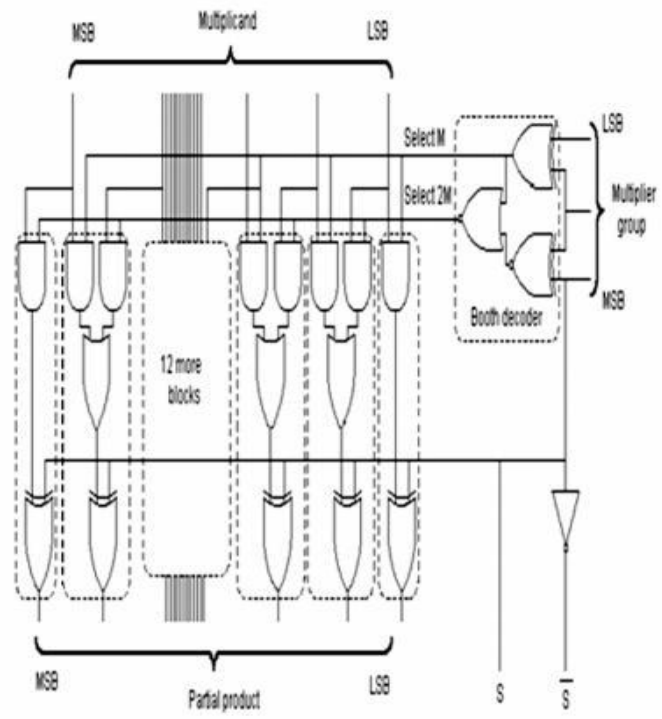


Figure 10: Booth Partial Product Selector Logic

(B) Applying the SPST on the Compression Tree

The proposed SPST -equipped multiplier is illustrated in figure 11. The PP generator generates five candidates of the partial products, i.e., {–2*A*,–*A*, 0, *A*, 2*A*}. These are then selected according to the Booth encoding results of the operand *B*. When the operand besides the Booth encoded one has a small absolute value, there are opportunities to reduce the spurious power dissipated in the compression tree.

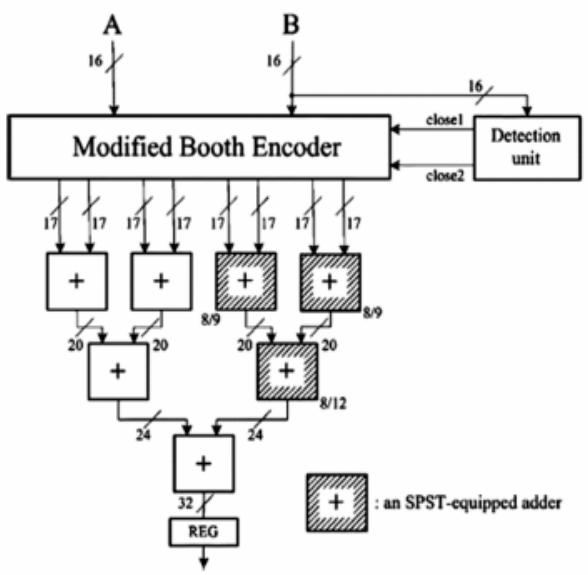


Figure11: Proposed High Performance Low Power Equipped Multiplier

VII. COMPARISION

The Proposed MAC results are very good when compared with the existig methods and the comparision table is given by following table.

Multiplier Type	**Array-MAC**	**MAC-CSA**	**MAC-SPST**
Vendor	Xilinx	Xilinx	Xilinx
Device & Family	Spartan3E XC3s1600e	Spartan3E XC3s1600e	Spartan3E XC3s1600e
Delay	298.974ns	55.916ns	53.805ns
Power Dissipation	123mW	123mW	16mW

VIII. SIMULATION RESULTS

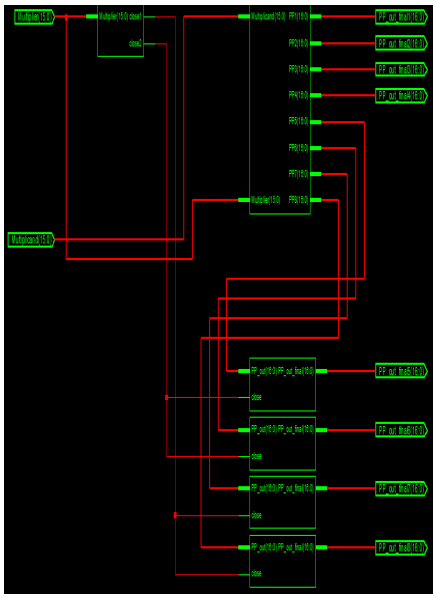


Figure 12: Schematic View of Booth Encoder

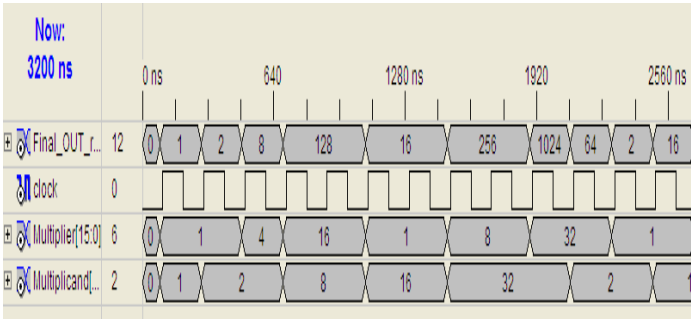


Figure 13: Simulation Waveform of Booth Encoder

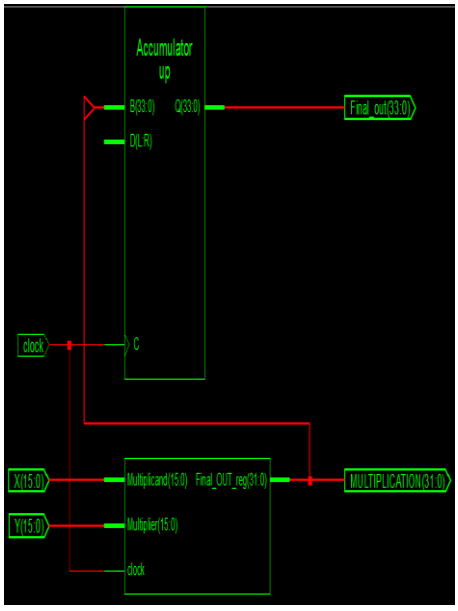


Figure 14: Schematic View of MAC

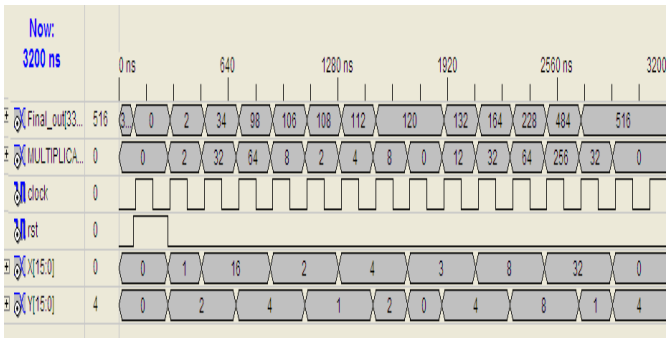


Figure 15: Simulation Waveform of MAC

IX. CONCLUSIONS

In this project, we propose a high speed low-power multiplier and accumulator (MAC) adopting the newSPST implementing approach. This MAC is designed by equipping the Spurious Power Suppression Technique (SPST) on a modified Booth encoder which is controlled by a detection unit using an AND gate. The modifiedbooth encoder will reduce the number of partial products generated by a factor of 2. The SPST adder will avoid the unwanted addition and thus minimize the switching power dissipation. The SPST MAC implementation with AND gates have an extremely high flexibility on adjusting the data asserting time. This facilitates the robustness of SPST can attain 30% speed improvement and 22% power reduction in the modified booth encoder. This design can be verified using Modelsim and Xilinx using verilog.

REFERENCES

[1] G. Lakshmi Narayanan and B. Venkataramani, "Optimization Techniques for FPGA-Based Wave Pipelined DSP Blocks", *IEEE Trans. Very Large Scale Integr. (VLSI) Syst*., **13.** No 7. pp 783-792, July 2005.

[2] L.Benini, G.D. Micheli, A. Macii, E. Macii, M. Poncino, and R.Scarsi,""Glitching Power Minimization by Selective Gate Freezing", *IEEE Trans. Very Large Scale Integr. (VLSI) Syst*., **8,** No. 3, pp. 287-297, June 2000.

[3] S.Henzler, G. Georgakos, J. Berthold, and D. Schmitt-Landsiedel, "Fastpower-Efficientcircuit-Block Switch-off Scheme", *Electron. Lett.*, **40,** No. 2, pp. 103-104, Jan. 2004.

[4] H. Lee, "A Power-Aware Scalable Pipelined Booth Multiplier", *In Proc. IEEE Int. SOC Conf*., 2004, pp. 123-126.

[5] J. Choi, J. Jeon, and K. Choi, "Power Minimization of Functional units by Partially Guarded Computation", *In Proc. IEEE Int. Symp*. Low Power Electron. Des., 2000, pp. 131-136.

[6] Z. Huang and M. D. Ercegovac, "High-Performance

Low-Power Left-Toright Array Multiplier Design", *IEEE Trans. Comput*., **54,** No. 3, pp. 272-283, Mar. 2005.

[7] K. H. Chen, K. C. Chao, J. I. Guo, J. S. Wang, and Y.S. Chu, "An Efficient Spurious Power Suppression Technique (SPST) and its Applications on MPEG-4 AVC/H.264 Transform Coding Design", *InProc. IEEE Int. Symp. Low Power Electron. Des.*, 2005, pp. 155-160.

[8] K. H. Chen, Y. M. Chen, and Y. S. Chu, "A Versatile Multimedia Functional Unit Design using the Spurious Power Suppression Technique", *in Proc. IEEE Asian Solid-State Circuits Conf.*, 2006, *pp.* 111-114.

[9] Zhijun Huang,"High Level Optimization Techniques for Low Power Multiplier Design" University of California, los angels, 2003.