International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064

# Modified Booth Multiplier with FIR Filter

**B. Sireesha[1], Diana Aloshius[2]**

[1,2]Department of ECE, Hindustan University, Chennai, Tamil Nadu, India

**Abstract:** ***In this paper, we develop a new methodology for designing a lower-error and area efficient 2's complement fixed-width Booth multiplier that receives two n-bit numbers and produces an n-bit product. By properly choosing the generalized index and binary thresholding, we derive a better error-compensation bias to reduce the truncation error. Since the proposed error compensation bias is realizable, constructing low-error fixed width Booth multiplier is area and time efficient for VLSI implementation. Finally, we successfully apply the proposed fixed-width Booth multiplier to FIR filter. The simulation results show that the performance is superior to by using the direct-truncation fixed-width Booth multiplier.***

**Keywords:** Modified Booth Multiplier, Booth Encoder, partial product, FIR, Signed-unsigned.

## 1. Introduction

Booth's multiplication algorithm is an algorithm which multiplies two signed binary numbers in two's complement notation. The algorithm was invented by Andrew Donald Booth in 1950 while doing research on crystallography at Birkbeck College in Bloomsbury, London. Booth used desk calculators that were faster at shifting than adding and created the algorithm to increase their speed. Booth's algorithm is of interest in the study of computer architecture.It can be used for both signed-magnitude numbers as well as 2's complement numbers with no need for a correction term or a correction step

This technique has the advantage of reducing the number of partial products by one half regardless of inputs. The recoding is performed within two steps: encoding and selection. The purpose of the encoding is to scan the triplet of bits of the multiplier and define the operation to be performed on the multiplicand, as shown in fig.1. This method is actually an application of a sign-digit representation in radix-4 [1]. The Booth-MacSorley algorithm, usually called the Modified booth algorithm or simply the booth algorithm. For example, a 3-bit recording would require the following set of digits to be multiplied by the multiplicand: 0, ±1, ±2, ±3. The difficulty lies in the fact that ±3Y is computed by summing (or subtracting) 1 to ±2Y, which means that a carry propagation occurs. The delay caused by the carry propagation renders this scheme to be slower than a conventional one. Consequently, only the 2 bit Booth recording is used.

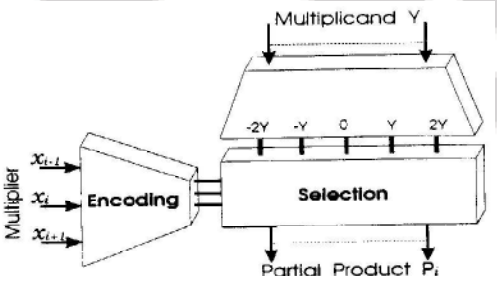


**Figure 1:** Implementation of modified booth recording

## 2. Conventional Modified Booth Multiplier

Booth's algorithm examines adjacent pairs of bits of the *N*-bit multiplier *Y* in signed two's complement representation, including an implicit bit below the least significant bit, $y_{-1}$ = 0. For each bit $y_i$ for *i* running from 0 to *N*-1, the bits $y_i$ and $y_{i-1}$ are considered. Where these two bits are equal, the product accumulator *P* is left unchanged. Where $y_i$ = 0 and $y_{i-1}$ = 1, the multiplicand times 2*i* is added to *P*; and where $y_i$ = 1 and $y_{i-1}$ = 0, the multiplicand times 2*i* is subtracted from *P*. The final value of *P* is the signed product . Encoder and partial product generation circuit is shown in 2(a) and 2(b), corresponding truth table in table1.

The representation of the multiplicand and product are not specified; typically, these are both also in two's complement representation, like the multiplier, but any number system that supports addition and subtraction will work as well. As stated here, the order of the steps is not determined. Typically, it proceeds from LSB to MSB, starting at *i* = 0; the multiplication by 2*i* is then typically replaced by incremental shifting of the *P* accumulator to the right between steps; low bits can be shifted out, and subsequent additions and subtractions can then be done just on the highest *N* bits of *P* [1].

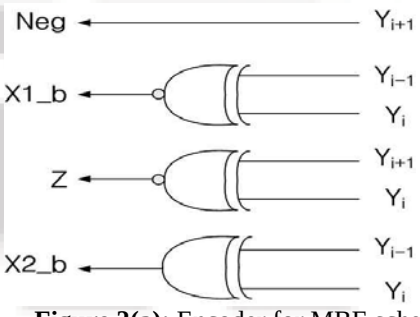


**Figure 2(a):** Encoder for MBE scheme

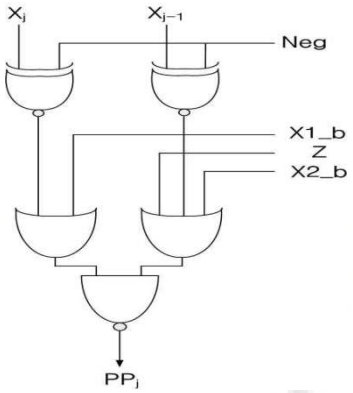


**Figure 2(b):** Decoder for MBE scheme

**Table 1:** Truth Table for Booth Encoder

| $Y_{N+1}$ | $Y_N$ | $Y_{N-1}$ | Operation | X | 2X | Neg |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | +0 x X | 0 | 0 | 0 |
| 0 | 0 | 1 | +1 x X | 1 | 0 | 0 |
| 0 | 1 | 0 | +1 x X | 1 | 0 | 0 |
| 0 | 1 | 1 | +2 x X | 0 | 1 | 0 |
| 1 | 0 | 0 | -2 x X | 0 | 1 | 1 |
| 1 | 0 | 1 | -1 x X | 1 | 0 | 1 |
| 1 | 1 | 0 | -1 x X | 1 | 0 | 1 |
| 1 | 1 | 1 | -0 x X | 0 | 0 | 1 |

## 3. Existing Method

Multipliers are key components of many high performance systems such as FIR filters, microprocessors, digital signal processors, etc. A system s performance is generally determined by the performance of the multiplier because the multiplier is generally the slowest clement in the system. Furthermore, it is generally the most area consuming. Hence, optimizing the speed and area of the multiplier is a major design issue. However, area and speed are usually conflicting constraints so that improving speed results mostly in larger areas. In parallel multipliers, high performance can be achieved by using modified Booth encoding, which decreases the number of partial products by a factor of 2 through multiplier recording.

Fixed word size in lossy systems that allow small accuracy loss to output data, can be maintained by using n×n fixed width multipliers which result in only *n* most significant product bits. Complexity reduction in hardware and significant area and power savings can be achieved by removing the adder cells of standard multiplier for the computation of the *n* less significant bits of *2n*-bit output product. However, the fallout being high truncation error will be introduced into the system to such kind of direct-truncated fixed-width multiplier. Variety of error compensation methods that add estimated compensation value to inputs of the reserved adder cells, which effectively arrests the truncation error. As it is universally known that error compensation value can be derived by the Adaptive/Constant scheme [2]. Regardless of current input data value influence, the Constant Scheme arrives to constant error compensation value and same is used to carry inputs of the retained adder cells while performing multiplication operations. Less hardware results in a relatively large truncation error to the Constant Scheme.

Adaptive scheme has been designed such that high accuracy is achieved by adjusting the compensation value according to the input data at the cost of higher hardware complexity. However, most of the adaptive error compensations are developed only for Fixed-width array multipliers & has less influence on reduction of truncation error for a fixed width modified booth multipliers directly. Various error compensation approaches, have been proposed to effectively scale down the truncation errors of Fixed-width modified Booth Multiplier at the cost of hardware complexity. Simple error compensation circuit results in better error performance and area with booth encoded outputs as inputs to generate the error compensation value.

The Modified Booth multiplier is an extension of Booth's multiplier. In Modified Booth, the number of partial products reduced by N/2, that is half of total partial products as compare to simple multiplication process [3]. So, clearly if the number of partial products becomes reduced, the area of the multiplier also will reduce and automatically as the result of it, the speed will increased. So, this multiplier is more efficient and Comparison shown in table 2.

**Table 2:** Comparison of sign-magnitude number multiplication with and without booth encoding

| *Booth encoding* | *No Booth encoding* |
|---|---|
| Internal Representation: 2's complement (some partial products need to be subtracted ) | Internal Representation: Sign magnitude (all the partial products are positive) |
| Hardware for encoding and selection | One row of 4:2 compressor |
| Sign extension | Only one XOR is used to compute the sign in parallel |
| 2 extra bits (sign extension and complementation) | No extra bits |
| The normalization requires some Leading Zero Detectors and Leading One Detectors | The normalization and even the rounding is easy |
| The schematic and the layout are not regular | The simplicity of the schematic allows a highly regular layout |
| 1 XOR + 1 AND(encoding), 2 XOR+1 AND (multiplexer) Total: 3 XOR +2 AND | 1 AND (partial product generation) ,3 XOR (4:2 compressor) Total: 3 XOR +1 AND |

### 3.1 Modified Booth Encoder (MBE)

Modified Booth encoding is most often used to avoid variable size partial product arrays. Before designing a MBE, the multiplier B has to be converted into a Radix-4 number by dividing them into three digits respectively according to Booth Encoder, Prior to convert the multiplier, a zero is appended into the Least Significant Bit (LSB) of the multiplier [4]. Table 3 shows the truth table for a booth encoder with two's, one's and correction bit [4]-[6].

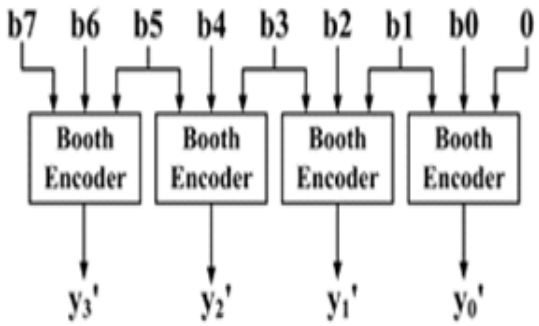


**Figure 3:** Grouping of bits for MBE scheme

The operand that is booth encoded is called multiplier, and the other operand is called multiplicand. If a 3-bit binary input sequence is given at the input, and perform the operation as mentioned in front of it [4], the partial products will be generated. As mentioned in fig.3, there are total 3 combinations for generating a partial product i.e., the obtained partial product is dependent upon the combination of three binary bits of the multiplier.

## 4. Proposed Method

In Booth multiplier Multiplication operation when considered between multiplicand and the multiplier namely A & B respectively, representation of which is depicted as below:

Let us consider the multiplication operation of two –bit signed numbers $A = a_{n-1}a_{n-2} \ldots \ldots a_o$(multiplicand) and $B = b_{n-1}b_{n-2} \ldots \ldots b_o$(multiplier). The two's complement representations of A and B can be expressed as follows:

$$A = -a_{n-1}2^{n-1} + \sum_{i=0}^{n-2} a_i\, 2^i,$$
$$B = -b_{n-1}2^{n-1} + \sum_{i=0}^{n-2} b_i\, 2^i \ldots \ldots \ldots (1)$$

Modified Booth encoding groups the multiplier bits into triplets, and are encoded as given in table II, then B can be expressed as:

$$B = \sum_{i=0}^{\frac{n}{2}-1} M_i 2^{2i} = \sum_{i=0}^{\frac{n}{2}-1} (-2b_{2i+1} + b_{2i} + b_{2i-1})2^{2i} \quad \ldots..(2)$$

Where b-1 =0 and $K_j$ belongs to the set of values {-2, -1, 0, 1, 2}.

According to the Modified Booth encoding table the existing booth encoder and partial product generation circuit are shown in figure 4 (a) and figure 4 (b) respectively.

With relation to negation operation, each bit of multiplicand *A* is complemented and an extra binary value „1" is added to least significant bit pertaining to next partial product row. To implement correction bit $C_j$ addition of, 1's being used and thereby indicating the partial product row $PP_j$ as positive ($C_j$=0) or negative ($C_j$=1). As each partial product row is represented in two s complementation, the sign bit for $PP_j$ each *must* be properly extended up to the $(2n-1)^{th}$bit position. Many ways [14] are proposed to give a solution to eliminate the problem of sign extension bits as they affect the performance and power consumption of the parallel multiplier with large values of *n*. The partial product matrix of an 88 modified booth multiplier with sign extension elimination technique is illustrated in Fig 5, where in $P_{j,k}$ and $S_j$ denote the $K^{th}$ product bit and the sign bit of partial product row $PP_j$, respectively.

**Table 3:** Modified booth encoding table

| Booth Encoder i/p $b_{2j+1}b_{2j}b_{2j-1}$ | Operation | Booth encoder o/p $N_j\ t_jO_jZ_jC_j$ |
|---|---|---|
| 0 0 0 | +0 | 0 0 0 1 0 |
| 0 0 1 | +A | 0 0 1 0 0 |
| 0 1 0 | +A | 0 0 1 0 0 |
| 0 1 1 | +2A | 0 1 0 0 0 |
| 1 0 0 | -2A | 1 1 0 0 1 |
| 1 0 1 | -A | 1 0 1 0 1 |
| 1 1 0 | -A | 1 0 1 0 1 |
| 1 1 1 | -0 | 1 0 0 1 0 |

### 4.1 Fixed-Width Modified Booth Multiplier

In the post-truncated modified Booth multiplier (PTM) the with the addition of an extra „1"(carry value by $LSP_{minor}$) at the (n-1)th bit position of partial product matrix as depicted in Fig 6 (including"1")then outputs the highly significant *n* product bits.

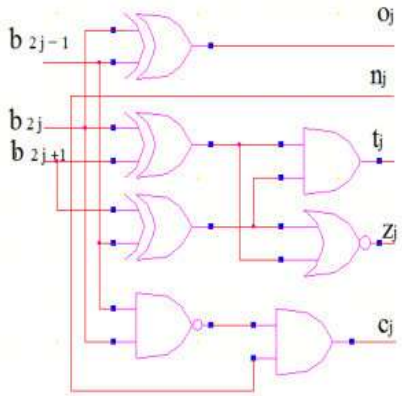


**Figure 4(a):** Modified Booth encoder

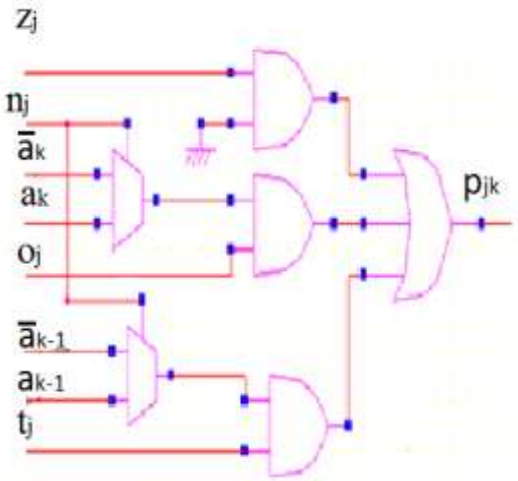


**Figure 4(b):** Partial product generation circuit

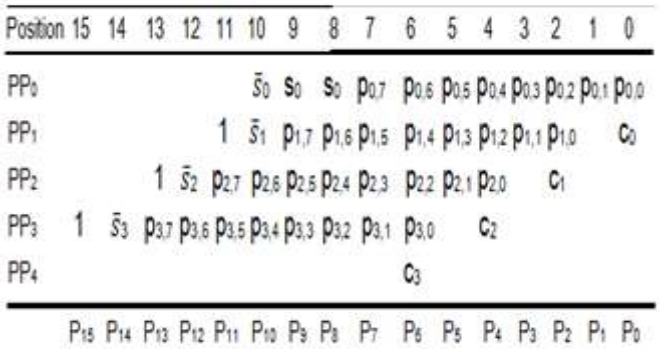


**Figure 5**: Partial product matrix of Booth multiplier

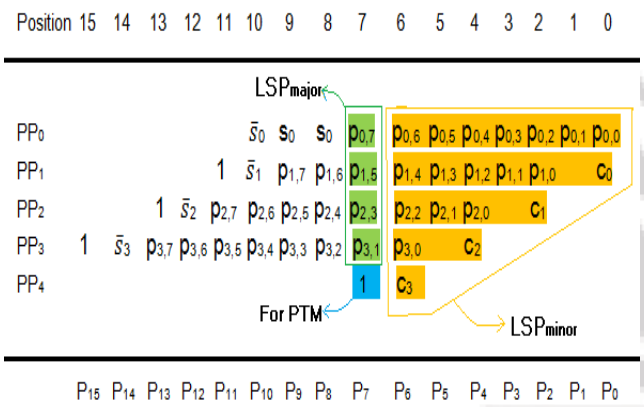


**Figure 6:** Partial product matrix of PTM

### 4.2 Multiplier in Fir Filter

As the complexity of digital filters is dominated by the number of multiplications, many works have focused on minimizing the complexity of multiplier blocks that compute the constant coefficient multiplications required in filters. The problem of designing FIR filters has received great attention during the last decade, as the filters are suffering from a large number of multiplications, leading to excessive area and power consumption even if implemented in full custom integrated circuits [8]. Early works have focused on replacing multiplications by decomposing them into simple operations such as addition, subtraction and shifting shown in Fig.7.

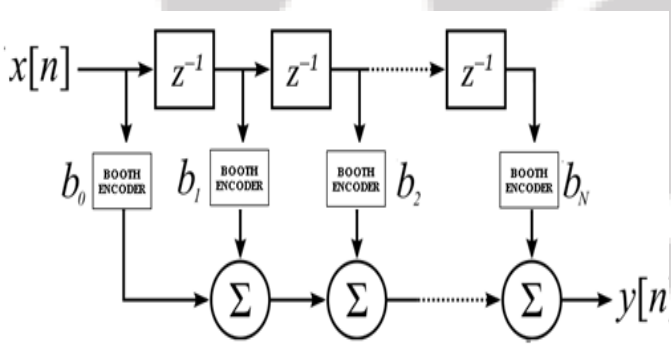


**Figure 7:** Multiplier In Fir Filter

As the coefficients of an application specific filter are constant, the decomposition is more efficient than employing multipliers. The complexity of FIR filters in this case is dominated by the number of additions/subtractions used to implement the coefficient multiplications.

## 5. Results and Observations

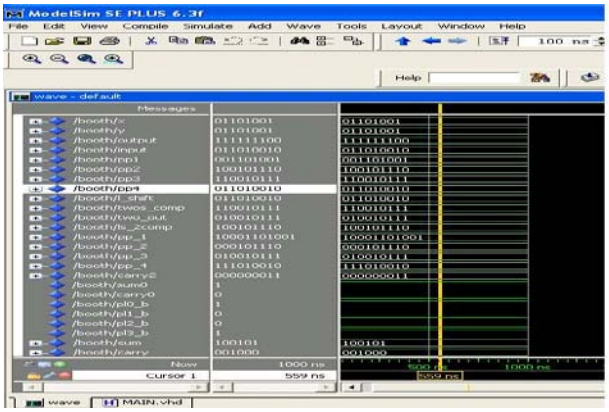

**Figure 8:** Conventional P-T booth

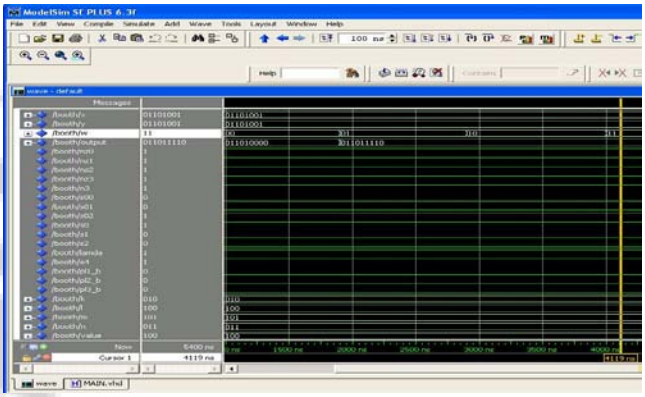

**Figure 9:** Modified booth multiplier

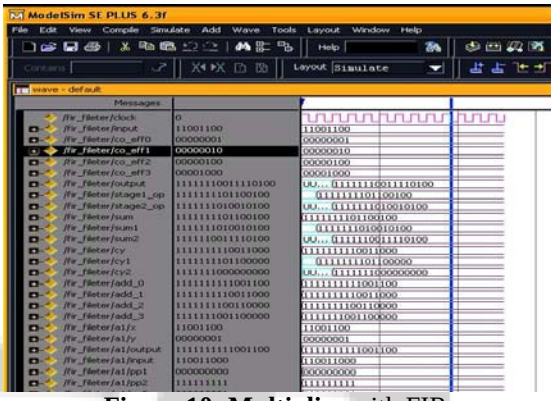

**Figure 10: Multiplier** with FIR

## 6. Conclusion

In this paper, we present a 8-bit×8bit advanced multiplier capable of carrying out both signed and unsigned operations. The proposed unified signed/unsigned multiplier was optimized in terms of speed, power consumption and silicon area by exploiting more regular partial product array, developing more efficient compression methods and combining several types of fast adders. Booth multiplier in FIR gives efficient results than FIR filter using array multiplier

## References

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

[2] G. O. Young, A. Inoue, R. Ohe, S. Kashiwakura, S. Mitarai, T. Tsuru,and T. Izawa, "A 4.1-ns compact 54 x 54 multiplier utilizing signs electBooth encoders," *IEEE J. Solid-State Circuits*, vol. 32, no. 11, pp. 1676–1682, Nov. 1997.
[3] K. Choi and M. Song, "Design of a high performance 32 x32-bit multiplier with a novel sign select Booth encoder," in *Proc. IEEE Int.Symp. Circuits and Systems*, 2001, vol. 2, pp. 701–704.
[4] S. S. Kidambi, F. El-Guibaly, and A. Antoniou, "Area-efficient multipliers for digital signal processing applications," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 43, no. 2, pp. 90–94, Feb. 1996.
[5] M. J. Schulte and E. E. Swartzlander, Jr., "Truncated multiplication with correction constant," in *Proc. VLSI Signal Processing, VI*, New York, 1993, pp. 388–396.
[6] L. D. Van and C. C. Yang, "Generalized low-error area-efficient fixed width multipliers," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 52, no. 8, pp. 1608–1619, Aug. 2005.
[7] J.-S.Wang, C.-N. Kuo, and T.-H. Yang, "Low-power fixed width array multipliers," in *Proc. Int. Symp. Low Power Electron. Des.* 2004, pp. 307–312.
[8] L. D. Van, S. S.Wang, and W. S. Feng, "Design of the low error fixed width multiplier and its application," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 47, no. 10, pp. 1112–1118, Oct. 2000.
[9] J. M. Jou, S. R. Kuang, and R. D. Chen, "Design of low-error fixed width multipliers for DSP applications," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process, vol. 46, no. 6, pp. 836–842, Jun. 1999.

## Author Profile

**B. Sireesha** obtained her B.Tech degree in Electronics and Communication Engineering from Don Bosco institute of technology and science, Guntur, Andhra Pradesh, India in 2012 and she is currently pursuing M.Tech in VLSI Design from Hindustan University, Chennai, Tamilnadu. Her field of interest includes high speed and low power VLSI architecture, Embedded Systems and DSP applications.

**Diana Aloshius** received her B.E. degree in Electronics and Instrumentation Engineering from Bharathiar University, India in 2004 and M.E. degree in Applied Electronics from Anna University, India in 2006. Currently, she is working as an Assistant Professor in the Department of Electronics and Communication Engineering, Hindustan University, Chennai. Her research interests include Digital Circuits and logic design, Low power VLSI and Reversible logic and synthesis.