IMPLEMENTATION OF 4 BIT BINARY MULTIPLIER USING WALLACE TREE MULTIPLICATION ALGORITHM AND A BRIEF COMPARISON WITH VEDIC AND CONVENTIONAL MULTIPLIER

Debmalya Paul, Kingsuk Das, Koyel Dey

1Department of Computer Science & Engineering Institute of Engineering & Management Kolkata, India
2Department of Electronics & Telecommunication Engineering Jadavpur University Kolkata, India
debmalyapaulfbp@gmail.com, kingsuk0405@gmail.com, deykoyel360@gmail.com

ABSTRACT

Multipliers are one of the major building block in most of the high performance digital systems such as Microprocessors, Signal processing Circuits, FIR filters etc. The demand of fast multipliers with low power consumption is increasing day by day. Inspired by this fact a 4 bit Wallace tree multiplier has been designed in this work. All the simulations has been carried out in VHDL version 9.1. Simulation result shows considerable improvement in terms of delay and memory utilization, as compared to Vedic multiplier and conventional multiplier. Thus, the superiority of the Wallace tree multiplier is established.

Keywords—Multiplier; Vedic multiplication algorithm; Wallace tree multiplication algorithm.

[1] INTRODUCTION

Power consumption, delay and area are three important design considerations for any chip designer. Multipliers play a major role in many of the DSP, FFT processors. Delay of the circuit is directly related to the delay of the multiplier and thus to the processor. Therefore research is going on to reduce the delay of multiplier so that the delay of whole circuit can be reduced. An early description of the Wallace tree multiplier was given by [1]. Wallace tree multiplier has been described as one of the most efficient multiplier.

In recent years a lot of research is going on for the improvement of multiplication technique. Researchers are working on various multiplication techniques such as Vedic multiplication, Wallace tree multiplication technique and trying to propose a better algorithm.

In [2], a 32 bit multiplier design was proposed using Wallace tree algorithm in which the architecture is based on modified Radix-4 Booth encoder, a modified Wallace Tree adder, and a Carry Look Ahead adder. The design was found to attain high speed, low delay, and low Silicon area. Also,
it performed better than booths algorithm which was considered as one of the fastest multiplication techniques.

In [3], Wallace tree multiplier has been designed using carry select adder with binary to excess-1 converter. The design was found to be quite area and power efficient.

In [4] and [5], a 8 bit and 16 bit binary multipliers are designed using multi channel CMOS (McCMOS) technology. In this work, channel length of NMOS and PMOS are different. And the entire design was found to be power efficient and leakage current proof.

In this work we have designed 4 bit binary multiplier using Wallace tree multiplication algorithm. To estimate the superiority of the aforementioned multiplier we have compared the simulation results with 4 bit Vedic multiplier and 4 bit Conventional multiplier. The comparison shows that the Wallace tree multiplier achieves much lesser amount of delay as compared to others.

The organization of the paper is as follows:

Section 2 contains Conventional multiplier technique, Section 3 deals with the Wallace tree multiplication algorithm, vedic multiplication technique is described in Section 4. Section 5 encompasses the simulation results and discussions.

**[2] CONVENTIONAL MULTIPLIER**

Conventional multiplication technique is one of the oldest multiplication techniques.

Fig. 1. n-bit conventional multiplier

Fig.1. describes the conventional multiplication method of two binary n bit numbers. In this case, the multipliers and multiplicands are denoted by ‘a’ and ‘b’ respectively, where i =0,1,2,…,n-1. First, we generate the partial products by multiplying multiplicands and multipliers. Addition of these partial products using half adders and full adders leads to generate the final results denoted by P_i, where i =0,1,2,…,n-1.

**[3] WALLACE TREE MULTIPLICATION**

Wallace tree multiplication algorithm is divided into three major steps-
1. Generation of partial products
2. Addition of partial products
3. Generation of final results from the partial products

Fig. 2. 4 bit Wallace tree multiplication method

In this case, the multipliers and multiplicands are denoted by ‘ai’ and ‘bi’ respectively, where i = 0, 1,..,3 for 4 bit multiplier. At first, the partial products are generated by multiplying multiplicands and multipliers. Partial products present in first three successive rows are added using half adders and full adders to generate ri, where i = 0, 1, 2,..,5. Ci, ri, and the partial products present in fourth row are further added to generate the sum pi, where i = 0, 1, 2,..6. Carry ci and pi are added in the final step to generate Sum and carry which are denoted by Si and C6, where i = 0, 1,..,6.

The architectural design of 4 bit Wallace bit tree multiplier is shown fig. 3. Generation of partial products are done using AND gate block. Output of all 16 AND gates are partial products. AND gate block is designed using VHDL data flow model. These partial products are fed to half adder and full adder blocks, which generates the sum and carry. Half adder and full adder blocks consist of XOR, AND, OR gates respectively. These gates are also designed using VHDL data flow model.
Vedic multiplication is the ancient technique for multiplication. Here, we have implemented the UrdhaTriyakbhyam algorithm for the multiplication of two 4 bit binary numbers. The binary numbers are denoted by $a_i$ and $b_i$ where, $i=0,1,...,3$ for 4 bit multiplier. The algorithm of UrdhaTriyakbhyam multiplier for two four bit binary numbers denoted by $A = (a_3 \ a_2 \ a_1 \ a_0)$ and $B = (b_3 \ b_2 \ b_1 \ b_0)$ is shown below.

**Step 1.** $a_3 \ a_2 \ a_1 \ a_0 \ b_3 \ b_2 \ b_1 \ b_0$  
$S_1 = a_0 \ AND \ b_0$.

**Step 2.** $a_3 \ a_2 \ a_1 \ a_0 \ b_3 \ b_2 \ b_1 \ b_0$  
$S_2 = (a_1 \ AND \ b_0) + (a_0 \ AND \ b_1)$.

**Carry 2**

**Step 3.** $a_3 \ a_2 \ a_1 \ a_0 \ b_3 \ b_2 \ b_1 \ b_0$  
$S_3 = (a_2 \ AND \ b_0) + (a_0 \ AND \ b_2) + (a_1 \ AND \ b_1)$.

**Carry 3**

**Step 4.** $a_3 \ a_2 \ a_1 \ a_0 \ b_3 \ b_2 \ b_1 \ b_0$  
$S_4 = (a_3 \ AND \ b_0) + (a_0 \ AND \ b_3) + (a_2 \ AND \ b_1) + (a_1 \ AND \ b_2)$.

**Carry 4**

**Step 5.** $a_3 \ a_2 \ a_1 \ a_0 \ b_3 \ b_2 \ b_1 \ b_0$  
$S_5 = (a_3 \ AND \ b_1) + (a_1 \ AND \ b_3) + (a_2 \ AND \ b_2)$.

**Carry 5**

**Step 6.** $a_3 \ a_2 \ a_1 \ a_0 \ b_3 \ b_2 \ b_1 \ b_0$  
$S_6 = (a_3 \ AND \ b_2) + (a_2 \ AND \ b_3)$.

**Carry 6**

**Step 7.** $a_3 \ a_2 \ a_1 \ a_0 \ b_3 \ b_2 \ b_1 \ b_0$  
$S_7 = a_3 \ AND \ b_3$.

**Carry 7**

Final result is denoted by $F_i$ where $i = 1,...,8$.  
$F_1 = S_1$.
Debmalya Paul, Kingsuk Das And Koyel Dey

\[ F2 = S2 \]
\[ F3 = S3 + \text{Carry 2 (Carry 3B)} \]
\[ F4 = S4 + \text{Carry 3 + Carry 3B (Carry 4B)} \]
\[ F5 = S5 + \text{Carry 4 + Carry 4B (Carry 5B)} \]
\[ F6 = S6 + \text{Carry 5 + Carry 5B (Carry 6B)} \]
\[ F7 = S7 + \text{Carry 6 + Carry 6B (Carry 7B)} \]
\[ F8 = \text{Carry 7 + Carry 7B (Carry 8B)} \]

The main advantage of Vedic Multiplication is that the numbers of steps are lesser as compared to the conventional which in turn reduces the delay and power consumption of Vedic multiplier.

**[5] SIMULATION RESULT AND DISCUSSION**

In this paper we have designed 4 bit binary multiplier using Wallace tree multiplication algorithm. We have also designed 4 bit Vedic multiplier and conventional multiplier. The output waveform of 4 bit Wallace tree multiplier is shown in fig.4. In fig.4. the inputs are defined as A and B, where \( A = a3 \ a2 \ a1 \ a0 \), \( B = b3 \ b2 \ b1 \ b0 \) and the final results are defined by \( S_i \) where, \( i = 0, 1, \ldots, 7 \).

![Output Waveform of 4 bit Wallace tree multiplier](image)

The detailed analysis of these simulations is shown in table1 and 2. All the simulations have been performed using Xilinx ISE 9.1. Table 1 shows that delay of Wallace tree multiplier is much lesser as compared to Conventional and Vedic multiplier. Memory usage of Wallace multiplication algorithm is also lesser as compared to conventional but equal to Vedic multiplier.

**TABLE 1. Detailed analysis of the three multipliers:**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Delay</td>
</tr>
<tr>
<td>Conventional</td>
<td>23.71</td>
</tr>
<tr>
<td>Wallace Tree</td>
<td>15.941</td>
</tr>
<tr>
<td>Vedic</td>
<td>17.589</td>
</tr>
</tbody>
</table>
TABLE 2. Percentage of improvement of the Wallace tree multipliers as compared to others:

<table>
<thead>
<tr>
<th>Wallace Tree Multiplier</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Improvement with respect to</td>
<td></td>
</tr>
<tr>
<td>Conventional</td>
<td>Delay</td>
</tr>
<tr>
<td></td>
<td>32%</td>
</tr>
<tr>
<td>Improvement with respect to Vedic</td>
<td>10%</td>
</tr>
</tbody>
</table>

The percentage of improvement of Wallace tree multiplier in terms of delay and memory is shown in table 2. It shows that delay of Wallace tree multiplier is 32% improved as compared to the conventional multiplier, whereas, the improvement is only 10% in case of Vedic multiplier. Also, the memory utilization is 1.10% less for Wallace tree multiplier as compared to conventional multiplier. It can be clearly seen from the tables that with the use of Wallace tree multiplication technique the performance of the multiplier can be considerably improved.

[6] CONCLUSION

In this paper we have designed and simulated a 4 bit Wallace tree multiplier. As seen from table 1, delay and memory utilization of Wallace tree multiplier is significantly less as compared to conventional multiplier. Though using the Wallace tree multiplication technique the memory utilization is almost same but the delay of the circuit reduces significantly as compared to the Vedic multiplier. We have designed the circuit keeping our concentration primarily on reduction of delay, and memory utilization, it has been successfully shown that by using the Wallace tree multiplication algorithm.

REFERENCES