A Distinctive Approach for Vedic-Based Squaring Circuit

Shamim Akhter
Department of Electronics and Communication Engineering
Jaypee Institute of Information Technology, NOIDA, India
shamim.akhter@jiit.ac.in

Saurabh Chaturvedi
Department of Electronics and Communication Engineering
Jaypee Institute of Information Technology, NOIDA, India
saurabh.chaturvedi@jiit.ac.in

Shaheen Khan
Department of Electronics and Communication Engineering
Mewat Engineering College
Haryana, India
shaheen.khan.2@gmail.com

Abstract - A novel method for squaring binary numbers using Vedic mathematics is proposed in this paper. The implementation of the binary squaring circuit uses the improved Vedic multiplier architecture. The circuit is designed in VHDL using the ModeSim tool from Mentor Graphics. The circuit synthesis is performed using the Xilinx ISE 14.1. The HDL-based simulation is presented for 4-bit and 8-bit. The design can easily be expanded for large bit sized inputs. The device utilization and delay comparison are presented using different families of FPGA. The binary squaring circuit presented in this paper show better speed performance than the previously reported squaring circuits.

Keywords - Vedic multiplier, Vedic squarer, carry save adder (CSA), increment by one (IBO), VHDL

I. INTRODUCTION

Vedic mathematics is a well-known technique which is used to improve the computational speed of arithmetic operations. Many Vedic based binary multipliers and squaring circuits are published in literature [1]-[9]. This paper presents a novel approach for the implementation of binary squaring circuits using an improved Vedic multiplier circuit.

The paper is organized as follows: Section presents the circuit topology and functionality of the formerly published 2-bit binary multiplier and squarer circuits. Section III proposes a new 2-bit binary squaring circuit. In Section IV, the Vedic technique is presented for multiplication [10]. Section V presents the proposed 4-bit Vedic squaring circuit. Section VI describes the simulation results of the squaring circuits designed using VHDL. This section also discusses the circuit synthesis results and performance comparison. Finally, the conclusion is drawn in Section VII.

II. 2-BIT MULTIPLIER AND SQUARER CIRCUITS

The block-level diagram of a 2-bit binary multiplier [1] is shown in Fig. 1, which uses two half adder (HA) blocks. In this figure, the 2-bit inputs are represented as \( A = A_1A_0 \) and \( B = B_1B_0 \), and output bits are \( R_2R_1R_0 \).

Fig. 1. Block diagram of 2-bit binary multiplier [1].

Using the above explanation, the 2-bit binary squarer circuit [1] is demonstrated in Fig. 2. This circuit uses one HA module and an AND gate, and the output is represented as \( P_3P_2P_1P_0 \).
III. PROPOSED MODIFIED 2-BIT BINARY SQUARER

The 2-bit squarer circuit discussed in Section II is modified and presented in this section. The HA of Fig. 2 adds A1 and A1A0 bits. The outputs of this HA are given below:

Sum (i.e. P2) = A1 XOR (A1A0) = A1.(NOT A0)  
Carry (i.e. P3) = A1.A0

These output expressions suggest that HA can be replaced with NOT and AND gates. The proposed 2-bit binary squaring circuit with NOT and AND gates is depicted in Fig. 3.

IV. VEDIC-BASED 4-BIT MULTIPLICATION

For designing the squaring circuit proposed in this paper, the architecture proposed in [10] for Vedic-based multiplier has been used.

V. PROPOSED 4-BIT VEDIC SQUARING CIRCUIT

Using the concepts discussed in previous sections and reported in [10], the computation steps for the proposed 4-bit squarer using Vedic technique are given below with the input data A as A3A2A1A0:

\[
\begin{array}{cccc}
A_3 & A_2 & A_1 & A_0 \\
\times & A_3 & A_2 & A_1 & A_0 \\
\hline
(A_3 A_2) \times (A_1 A_0) & (A_1 A_0) \times (A_1 A_0) \\
(A_3 A_2) \times (A_1 A_0) & (A_1 A_0) \times (A_1 A_0)
\end{array}
\]

The product terms are generated using 2-bit squarer (for first and last stage partial products) and 2-bit multiplier (for middle stage) using the techniques discussed in Sections III and II, respectively. The outputs of the above computation are shown below:

\[
(A_3 A_2) \times (A_3 A_2) \text{ gives 4-bit output as: } S_3 S_2 S_1 S_0 \\
(A_3 A_2) \times (A_1 A_0) \text{ gives 4-bit output as: } S_3 S_2 S_1 S_0 \\
(A_1 A_0) \times (A_1 A_0) \text{ gives 4-bit output as: } S_0 S_0 S_0 S_0
\]

Using the concept demonstrated in circuit shown in Fig. 4, the expression shown above can be arranged as:

\[
S_{33} S_{23} \quad S_{12} S_{12} S_{01} S_{02} \quad '0' \quad S_0
\]
It can be seen that the middle stage has two same entries at the first and second rows, therefore the output from the middle stage can be represented as: $2 \times (S_{32}S_{22}S_{12}S_{02}) + '0'S_{03}S_{30}S_{20}$.

The factor of 2 in the output of the middle stage can be incorporated by shifting complete data one bit to the left side. Therefore, the output from the middle stage is equivalent to addition of $S_{32}S_{22}S_{12}S_{02}'0'$ and $'0'S_{03}S_{30}S_{20}$. So it does not require CSA and normal 4-bit full adder can be used for the addition of middle stage. Fig. 5 depicts the schematic of the suggested 4-bit squaring circuit based on Vedic technique. The control signal for the 2-bit IBO module is the output of OR gate. $Cout$ and $S_{32}$ are the inputs of OR gate.

The proposed technique has been used for the implementation of 8-bit Vedic squarer as well. The building blocks required for the design of Vedic squaring circuits are given in Table 1.

VI. RESULTS AND DISCUSSION

This section describes the simulation and synthesis results of the implemented Vedic squaring circuits using ModelSim and Xilinx platforms, respectively.

(a) Simulation waveforms for the proposed 4-bit Vedic squarer

For a 4-bit input $A$ represented by $a3a2a1a0$, the simulation waveforms for the 8-bit square output $S$ represented by $s7s6s5s4s3s2s1s0$ are demonstrated in Fig. 6.

For input $A=1010$, the output $S=01100100$

For input $A=1110$, the output $S=11000100$

For input $A=0111$, the output $S=00110001$

(b) Simulation waveforms for the proposed 8-bit Vedic squarer

The 8-bit input $A$ is $a7a6a5a4a3a2a1a0$ and 16-bit output $S$ is $s15s14s13s12s11s10s9s8s7s6s5s4s3s2s1s0$, as shown in Fig. 7.

For input $A=11111111$, the output $S=11111100000001$

For input $A=01101111$, the output $S=0011011101010001$

Synthesis is performed using different families of Xilinx FPGA kits. Table 2 compares the performance of the proposed and implemented Vedic squarer circuits with the previously published works. It can be observed that the proposed Vedic Squarer circuits are efficient compared to that published in literature.
TABLE II. Performance comparison of the Vedic squarer circuits

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Input size</td>
<td>8-bit</td>
<td>4-bit</td>
<td>8-bit</td>
<td>4-bit</td>
<td>8-bit</td>
</tr>
<tr>
<td>LUT</td>
<td>6</td>
<td>6</td>
<td>56</td>
<td>6</td>
<td>56</td>
</tr>
<tr>
<td>Delay (ns)</td>
<td>4.993</td>
<td>10.32</td>
<td>4.993</td>
<td>10.32</td>
<td>4.993</td>
</tr>
<tr>
<td>Spartan-3: XC3S50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LUT</td>
<td>6</td>
<td>6</td>
<td>56</td>
<td>6</td>
<td>56</td>
</tr>
<tr>
<td>Delay (ns)</td>
<td>7.86</td>
<td>18.67</td>
<td>9.07</td>
<td>19.8</td>
<td>17.3</td>
</tr>
<tr>
<td>Spartan-6: XC6SLX150T</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LUT</td>
<td>6</td>
<td>6</td>
<td>56</td>
<td>6</td>
<td>56</td>
</tr>
<tr>
<td>Delay (ns)</td>
<td>4.99</td>
<td>10.32</td>
<td>6.76</td>
<td>56</td>
<td></td>
</tr>
</tbody>
</table>

Delay (ns) 4.99 10.32 6.76 56
LUT 6 6 6 6
Delay (ns) 4.99 10.32 6.76 56

NR: Not reported in corresponding reference

From Table 2, using Spartan-3: XC3S50, the proposed 4-bit Vedic squarer shows 13.34% and 54.57% less delay compared to the Vedic squaring circuits of [4] and [5], respectively. The proposed 8-bit Vedic squarer displays 5.71% less delay presented in [6]. Using Virtex-4: XC4VLX15, the designed 8-bit squaring circuit exhibits 27.63% less delay than that of [9].

With the reported synthesis results, it is concluded that the proposed Vedic-based binary squaring circuits are more efficient and can be used for high-performance computational applications.

VII. CONCLUSION

This paper presents a novel Vedic-based approach for the design of binary squaring circuits. The synthesis results demonstrate that the implemented squaring circuits are more efficient in terms of delay. The discussed design approach can be expanded and applied for large input data, such as 16-bit, 32-bit, and 64-bit. Some other multiplier circuits [11], [12] can also be used in the implementation of squaring operation. Also different types of adders as discussed in [13]-[15] can also be used to check the performance parameter of the proposed Vedic squarer. The proposed multiplier can also be used to enhance the performance of mathematical operations in residue number system [16]-[18] and FIR filter application [19].

REFERENCES