A PROPOSED APPROACH FOR APPROXIMATE COMPRESSORS BASED MULTIPLICATION
Ankita Leela¹, K.K.V.Girija²
¹Department of Electronics and Communication Engineering, Swarnandhra College of Engineering and Technology, Narsapur, (West Godavari), Andhra Pradesh, India.
²Department of Electronics and Communication Engineering, Swarnandhra College of Engineering and Technology, Narsapur, (West Godavari), Andhra Pradesh, India.

KEYWORDS: Omission Flexible, Inferred, Compressors, Dadda Multiplier, CMOS.

ABSTRACT
The most engaging model for PC calculating applications is estimated figuring omission of flexible applications, for instance, and blending media propelled sign planning. The arrangement and examination of two new inferred compressors is presented in this unique replica. The incorrect enlisting presents imprecision in estimation. Be that as it may, it conveys fast results with low power use. Four exceptional arrangements for utilizing the cautious compressors and wrong compressors are dismembered for a dadda multiplier. The distraction results in identifying that, derived multipliers fulfill diminishments in power usage, delay, transistor number, when differentiated and exact multipliers through hand specialty and organization in 0.5µm CMOS process advancement. The execution of the distinct and proposed structures is surveyed using aide representation instrument.

INTRODUCTION
The Most of the PC math applications are executed utilizing accurate, exact and deterministic calculations. Be that as it may, the ability to persevere through the blunders, produce significant and valuable results is feasible for applications like sight and sound and advanced picture preparing [1]. For such applications, the use of deterministic and careful models is not regularly productive and suitable [2]. The model of inaccurate registering depends on the property to divert the current configuration procedure of advanced frameworks by exploiting a diminishing in multifaceted nature and expense with conceivably an upgrade in execution and force proficiency [3].

Multipliers and adders structure avital equipment obstruct in the computerized sign handling and implanted applications. For summation, full adders have been broke down and various surmised plans have been proposed. Increase velocity decides processor speed [4]. So fast multipliers are required in the processors for some applications. For expanding the rate of increase, distinctive calculations are utilized [5].

Duplication is a most generally utilized operation as a part of numerous figuring frameworks [6]. A number (multiplicand) is added to itself various times, as determined by another number (multiplier) to shape an outcome (item). Be that as it may, the execution of multiplier takes tremendous equipment assets and the circuit works at low speed [7]. Compressors are broadly used to facilitate the incomplete item diminishment tree in the outline of a quick multiplier. In the late days, inexact multipliers increased critical significance [8].

At first in this paper, two new estimated 4-2 compressors have been proposed and investigated [1] [9]. These compressors have preferable postpone and force utilization over the precise 4-2 compressors [10]. The definite compressors and additionally surmised compressors are utilized as a part of the compensation module of a dadda multiplier [11]-[15].

The association of paper is as per the following. Area II presents the new plans of 4-2 vague compressors notwithstanding the current's audit outline techniques of compressors. Area III presents the four multipliers. The reenactment results for the compressors and multipliers are spoken to in segment IV. Area V shows the composition's finish.

DESIGN METHODOLOGIES
EXACT COMPRESSIONS
The main objective of multi-operand carry-save addition or parallel multiplication is the reduction of n numbers to two numbers. A slice of circuit that performs this reduction when properly replicated is a n-2 compressor (figure 1). The following inequality must be satisfied for the correct operation of the circuit represented in figure 1.
\[ n\Psi_1 + \Psi_2 + \Psi_3 + \ldots \leq 3\Psi_1 + 4\Psi_2 + 8\Psi_3 + \ldots \quad (1) \]

Figure 1. Schematic diagram of \( n \)-2 compressors in a multi operand adder circuit

An exact 4-2 compressor (figure 2) produces two outputs sum and carry when four inputs are applied. A carry (\( C_{\text{out}} \)) is generated, which acts as carry in for the next higher order compressor. A 4-2 compressor can be implemented using two full adders (figure 3).

Figure 2. An accurate compressor

Figure 3. Implementation of 4-2 Compressor
The optimized design of a 4-2 compressor based on XOR-XNOR gates is shown in figure 4. Both XOR and XNOR outputs are obtained from this gate. The design of this compressor comprises three XOR-XNOR gates, one XOR gate and two multiplexers. The truth table of an exact compressor is shown in table I. The logic equations of the exact compressor are as follows:

\[
\begin{align*}
\text{Sum} &= X_1 \oplus X_2 \oplus X_3 \oplus X_4 \oplus C_{in} \quad (2) \\
C_{out} &= (X_1 \oplus X_2) X_3 + (X_1 \oplus X_2) X_1 \quad (3) \\
\text{Carry} &= (X_1 \oplus X_2 \oplus X_3 \oplus X_4) C_{in} + (X_1 \oplus X_2 \oplus X_3 \oplus X_4) X_4 \quad (4)
\end{align*}
\]

**Figure 4. An optimized accurate compressor**

**TABLE I**

TRUTH TABLE OF 4-2 COMPRESSOR

<table>
<thead>
<tr>
<th>(C_{in})</th>
<th>(X_4)</th>
<th>(X_3)</th>
<th>(X_2)</th>
<th>(X_1)</th>
<th>(C_{out})</th>
<th>(carry)</th>
<th>(sum)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

APPROXIMATE COMPRESSORS

Two designs of inexact compressors are proposed in this manuscript. For the significant reduction of hardware, the truth table of the exact compressor is modified and the approximate designs have been proposed.

**Design 1 approximate compressor**

The carry output in an exact compressor has the same value of the input \( C_{in} \) in 24 out of 32 states as shown in the table 1. In this design, the carry is simplified to \( C_{in} \) by changing the value of other 8 outputs.

\[
\text{Carry} = C_{in} \quad (5)
\]

The logic expressions for the approximate sum and carry of design 1 inexact compressor are as follows

\[
\text{Sum} = \overline{C_{in}}(X_1 \oplus X_2 + X_3 \oplus X_4) \quad (6)
\]

\[
C_{out} = (X_1X_2 + X_3X_4) \quad (7)
\]

**TABLE II**

TRUTH TABLE OF THE DESIGN 1 APPROXIMATE COMPRESSOR

<table>
<thead>
<tr>
<th>( C_{in} )</th>
<th>( x_4 )</th>
<th>( x_3 )</th>
<th>( x_2 )</th>
<th>( x_1 )</th>
<th>( C_{out} )</th>
<th>carry</th>
<th>Sum</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Albeit the above mentioned simplifications of carry and sum enhance the error rate in the approximate compressor, the complexity and power consumption is reduced. This can be observed from comparison of equations (2)-(4) and (5)-(7). Also, the critical path delay in the proposed design is lower than in the exact design. The truth table of design 1 inexact compressor is shown in table II. The difference between the inexact output of the design 1 approximate compressor and the output of the exact compressor is shown in last column of the table II. As shown in Table II, an error rate of 37.5% is yielded in the proposed design, which has 12 incorrect outputs out of 32 outputs.

The gate level implementation of design 1 inaccurate compressor is shown in figure 5.

**Figure 5. Design 1 inaccurate compressor**

**Design 2 approximate compressor:**

In Design 2 inexact compressor, the C\textsubscript{in} and C\textsubscript{out} are completely ignored. This design achieves enhancement in performance as well as reduction in error rate. The proposed equations for the approximate carry and C\textsubscript{out} in the previous part can be interchanged since those outputs have the same weight. The logic expressions for the approximate sum and carry of design 2 inexact compressor are as follows

\[
\text{Sum} = (X_1 \oplus X_2 + X_3 \oplus X_4) \quad (8)
\]
\[
\text{Carry} = (X_1X_2 + X_3X_4) \quad (9)
\]

Equations (9) and (8) are same as (7) and (6) respectively for C\textsubscript{in} = 0.

The gate level implementation of design 2 inaccurate compressor is shown in figure 6.
The truth table of design 2 inexact compressor is shown in table III. The difference between the inexact decimal value of addition of outputs and the exact decimal value is shown in last column of the table III. The error rate is now reduced to 25% since this design has 4 incorrect outputs out of 16 outputs.

**TABLE III**

**TRUTH TABLE OF THE DESIGN 2 APPROXIMATE COMPRESSOR**

<table>
<thead>
<tr>
<th>x4</th>
<th>x3</th>
<th>x2</th>
<th>x1</th>
<th>Carry</th>
<th>Sum</th>
<th>Difference</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>-1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>-1</td>
<td>1</td>
</tr>
</tbody>
</table>

**MULTIPLICATION**

An exact multiplier usually comprises three modules.
- Partial product generation
- Partial Products Reduction Tree
- Redundant-to-Binary Converter

For designing a multiplier, the second module plays a vital role with respect to delay, power consumption and circuit complexity. Carry save adder is used to reduce the partial product matrix. Compressors are widely used to expedite the CSA. Four multipliers are implemented
- In the first case (Multiplier 1), exact compressor is used for partial product reduction as shown in Figure 8.
- In the second case (Multiplier 2), optimized exact compressor is used for partial product reduction as shown in Figure 10.
- In the third case (Multiplier 3), Design 1 approximate compressor is used for partial product reduction as shown in Figure 12.
In the fourth case (Multiplier 4), Design 2 approximate compressor is used for partial product reduction as shown in Figure 14.

The objectives of the two approximate designs are to reduce the delay, transistor count and power consumption when compared with the exact multipliers; however, a high error distance is expected in the former multipliers when compared to two latter multipliers.

**SIMULATION RESULTS**

In this area, the two's outlines estimated compressors (Section II) and the four inexact multipliers (Section III) are reproduced utilizing Eldo™ simulator. The schematics of every one of the four multipliers and formats of all compressors are planned in 0.5µm CMOS innovation utilizing mentor graphics tool. The compressors are simulated and are compared in table IV with respect to various metrics.

**TABLE IV
COMPARISON OF COMPRESSORS**

<table>
<thead>
<tr>
<th>Type of compressor</th>
<th>Transistor Count</th>
<th>Power Dissipation (nW)</th>
<th>Delay (n∆)</th>
<th>Error rate (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact</td>
<td>96</td>
<td>1.0696</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Optimized exact</td>
<td>64</td>
<td>0.985</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Design 1</td>
<td>60</td>
<td>0.642</td>
<td>3</td>
<td>37.5</td>
</tr>
<tr>
<td>Design 2</td>
<td>54</td>
<td>0.604</td>
<td>2</td>
<td>25</td>
</tr>
</tbody>
</table>

Figures 7, 8 and 9 represent the architecture and waveforms of 4*4 multipliers using an accurate compressor respectively.
Figure 8. Architecture of 4*4 multiplier using exact 4-2 compressor
Figures 10 and 11 represent the architecture and waveforms of $4 \times 4$ multipliers using an optimized accurate compressor respectively.
Figure 10. Architecture of 4x4 multiplier using optimized exact compressor
Figures 12 and 13 represent the architecture and waveforms of 4*4 multipliers using design 1 inaccurate compressor respectively.
Figure 12. Architecture of 4*4 multiplier using design 1 inaccurate compressor
Figures 14 and 15 represent the architecture and waveforms of 4*4 multipliers using design 2 inaccurate compressor respectively.
Figure 14. Architecture of 4*4 multiplier using design 2 inaccurate compressor
The four multipliers are simulated and they are compared in table IV with respect to delay, transistor count and power consumption and power-delay product.

**TABLE V COMPARISON OF MULTIPLIERS**

<table>
<thead>
<tr>
<th>Type of multiplier</th>
<th>Transistor Count</th>
<th>Power Dissipation (nW)</th>
<th>Delay(ns)</th>
<th>PDP(10^7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Multiplier using exact compressor</td>
<td>548</td>
<td>6.058</td>
<td>13.261</td>
<td>80.335</td>
</tr>
<tr>
<td>Multiplier using optimized exact compressor</td>
<td>516</td>
<td>5.954</td>
<td>13.258</td>
<td>78.938</td>
</tr>
<tr>
<td>Multiplier using design 1 inexact compressor</td>
<td>508</td>
<td>5.675</td>
<td>10.062</td>
<td>57.108</td>
</tr>
<tr>
<td>Multiplier using design 2 inexact compressor</td>
<td>506</td>
<td>5.402</td>
<td>7.5566</td>
<td>40.821</td>
</tr>
</tbody>
</table>
CONCLUSION
In this work, four multipliers have been proposed and are recreated utilizing Eldo™ test system utilizing ami 05 innovation. The execution is assessed utilizing coach representation apparatus. The formats of four compressors are composed and are contrasted with deference with postponement, power dispersal, and transistor number and slip rate.

The four multipliers are contrasted with deference with deferral, power dissemination, force delay item and transistor number. Among every one of the multipliers, multiplier actualized utilizing outline 2 estimated compressor has less power dissemination of 5.402 nW, delay 7.5566 ns and transistor number of 506. Notwithstanding, a huge lapse rate is accomplished in this multiplier. Inexact multipliers have less power scattering, postponement, force delay item and transistor check when contrasted with precise multipliers with a confinement of lapse rate.

REFERENCES


