Power and area efficient approximate carry skip adder for error resilient applications

Sujit Kumar PATEL1*, Bharat GARG1 and Shireesh Kumar RAI1

1Electronics and Communication Engineering, Thapar Institute of Engineering and Technology, Patiala, Punjab, India

Received: .201 • Accepted/Published Online: .201 • Final Version: .201

Abstract: The compute intensive multimedia applications on portable devices require power and area efficient arithmetic units. The adder is a prime building block of these arithmetic units and limits the overall performance. Therefore, this paper analyses logic operations of the state-of-the-art adders and presents a novel low complexity adder segment with new carry prediction logic by removing the redundant logic and sharing the common operations. Further, a new power and area efficient approximate carry skip (PAEA-CSK) adder is proposed using novel adder segment. The effectiveness of proposed PAEA-CSK adder is evaluated and compared over the existing adders by implementing them in VHDL and synthesizing using Synopsys Design Compiler with 65nm TSMC CMOS Library. The synthesis result shows that the proposed PAEA-CSK adder requires 27.28% and 18.03% less area and power respectively over the existing carry skip based approximate adder with same accuracy. Further, the Sobel edge detector (SED) embedded with proposed adder improves PSNR by minimum of 16.94 dB over the SED embedded with non-zeroing bit-truncation adder.

Key words: Approximate computing, carry skip adder, low-power design, error resilient applications

1. Introduction

The exponential growth of multimedia applications on modern portable devices demands highly power efficient and low complexity processing units. The conventional design techniques exhibit trade-off between area, power and delay where improving one metric degrades the other. Therefore, these design techniques fail to improve all the parameters simultaneously. Several applications, such as multimedia and big data analysis, exhibits error resiliency in abundance and do not requires 100% accurate results [1]. Relaxing the computational accuracy with acceptable quality may significantly reduce the area/power consumption. Therefore, approximate computing has emerged as new design paradigm for these applications and showed remarkable improvement in the performance. The accuracy is considered as new trade-off parameter in these applications to achieve improved design metrics. In various signal processing, the arithmetic unit consumes significant amount of area/power and determines the performance of the design. The adder is not only the key arithmetic unit in different signal processing; it is also used to implement other units such as multiplier, divider, incrementer, decremener etc. Therefore, design of power and area efficient adder is the critical requirement to achieve efficient signal processing systems.

Several design techniques have been reported to achieve high performance adder architectures. These techniques can be classified based on the computational accuracy, namely: accurate, approximate and accuracy-configurable design. The accurate design techniques exploit resource sharing and/or redundant logic elimination.

*Correspondence: sujit.patel@thapar.edu
Whereas, approximate design techniques identify and compute sum of non-significant part using approximate computing [2]. The accuracy-configurable designs are nothing but approximate designs with error detection and correction (EDC) logic to achieve desired accuracy-performance trade-off. In accurate adder designs, the ripple carry adder (RCA) is the simplest design but exhibits large carry propagation delay. Therefore, several high speed adders are presented in the literature [3]. For example, the carry look-ahead (CLA) adder computes carry signals in advance to reduce delay but shows large area overhead. Further, the carry select adder (CSLA) [4, 5] and carry skip adder (CSKA) [6] provide reduced delay with additional area and power consumption. On the other hand, to achieve area and power efficient design, the approximate adders are implemented either using simplified full adder [7] or by designing approximate sum logic for few least significant bits (LSBs) [8, 9]. The approximate sum logic is designed such that overall error is minimized. Further, a generic approximate carry skip adder (ACSKA) where previous $v$-segments are used to predict the carry-in for the current segment is presented in [10]. The consideration of more segments for carry prediction improves the accuracy at the cost of large delay and area. However the approximate adders provide improved design metrics with acceptable amount of error, they require large redesign efforts to achieve an approximate designs with different accuracy-performance trade-off.

There are several applications which require different accuracy under different conditions, for example, a security camera requires more clear images once the object is detected [11]. For these applications, accuracy of the adder should be reconfigurable during run-time to achieve maximum power benefits with acceptable quality. An accuracy configurable adder (ACA) [11] computes approximate sum using sub-adders and provides accurate result by adding error value in next pipelined stages. Therefore, it provides trade-off between computational latency and accuracy. A reconfigurable approximate CLA (RAP-CLA) adder [12] considers fixed $w$ terms (window) of carry expressions (conventional CLA) for carry generation during approximate mode while conventional CLA’s carry expressions in accurate mode. Although, the RAP-CLA shows high performance, it requires large power and area overhead due to the use of multiplexers for switching between accurate and approximate mode. Finally, a scalable non-zeroing bit-truncation (NzBt) adder presented in [13] which can switch between accurate and approximate mode using control signal. In approximate mode, the NzBt fixes certain LSB of input operands to complementary values which leads to output sum bits with constant logic ‘1’.

Among the recent approximate adders, the approximate carry skip adder provides better trade-off between area, delay and power performance. However, the existing carry skip (CSK) based approximate adder exhibits redundant logics in its design. Therefore, it could be possible to develop area and power efficient CSK based approximate adder by identifying and eliminating redundant logics. The key contributions are as follows:

- An analysis of state-of-the-art carry skip adders on the basis of logic complexity is presented.
- A novel low-complexity approximate adder segment with new carry prediction logic is proposed by removing the redundant logic operations.
- A new power and area efficient approximate carry skip (PAEA-CSK) adder is presented using the proposed adder segments.

Rest of the paper is organized as follows: Section 2 presents related work to achieve low power approximate/accuracy-configurable adders. Section 3 presents a comprehensive analysis of existing adders based on the logic complexity. The low complexity CSK adder segment is presented in Section 4 whereas, proposed area and power efficient adder is presented in Section 5. Section 6 presents simulation environment and comparative analysis of the proposed adder over the existing. Finally, Section 7 concludes the paper.
2. Related work

Low complexity arithmetic unit is the essential requirement of all modern devices to achieve efficient signal processing. Since adder is the basic building block of any arithmetic unit, several accuracy configurable adder architectures are reported in the literature. Based on the computational accuracy, review is broadly divided into three categories namely: accurate, approximate and accuracy-configurable adders, and is presented in the following subsections.

2.1. Accurate adder architectures

Among the several accurate adder architectures, RCA is the simplest and shows linear area and delay characteristic with bit-width. The worst case delay occurs when carry propagates from LSB to most significant bit (MSB) making this design as slowest one. Various high speed adders namely: CLA, CSKA and CSLA are presented to reduce the delay. The CLA pre-computes carry-in signals for each intermediate stage of addition and thus eliminates carry propagation. The combinational logic for generating carry-in is bulky for more than four bits. The CSKA reduces the carry propagation delay by implementing architecture such that input carry will be either killed in the segment or skipped when propagate condition is asserted. In the CSLA, partial sum is computed under the assumption of carry-in as logic ‘0’ and ‘1’. The carry-in signal is used to select correct sum and output carry instead of computing it, thus, significantly reducing the delay of adder. However several redundant logic elimination techniques are presented to reduce area of these adders [4–6], the area overhead of these high speed adders is very large making these designs area/power inefficient. Following subsection presents approximate adders that leverage relaxation in computational accuracy to achieve power/area efficient designs.

2.2. Approximate adder architectures

Most of the approximate adders are implemented by either using approximate full adder (FA) or employing specific logic to generate approximate sum for few LSBs. In [7], five different approximate FAs are presented by selectively eliminating few transistors from accurate mirror adders. In [14], XOR/XNOR based approximate FAs with reduced node capacitances and power consumption are presented. These FAs are then used at few LSBs to design large bit-width approximate adder. On the other hand, Zhu et al. [8] presented an error tolerant adder (ETA) where inputs are divided into two parts: upper part containing few MSBs whereas lower part containing remaining LSBs. The conventional accurate adder is used to compute sum of MSBs to reduce amount of error while simplified logic is used to compute sum of LSBs. It reduces the power and delay significantly at the cost of large error when small inputs are added. In [9], some OR gates compute approximate sum for the lower part to reduce the delay at the cost of large error. To reduce the amount of error, a lower part constant-OR adder (LOCA) is presented in [15]. The LOCA systematically computes approximate sum by employing constant logic ‘1’ for few LSBs, OR gates for few next MSBs and an accurate logic for the remaining MSBs. Recently, a reverse carry propagate FA is presented in [16] where significance of the carry-in is more than the carry-out. A process tolerant adder is presented in [17] that provides low power and high performance even under high process variation. Along with these techniques, various segment based approximate adders are also presented. These adders compute approximate sum for each segment using sub-adders and consider carry-in for each sub-adder to few previous segment(s). ETA-II, ETA-IIM [18] and carry skip approximate adders [10, 19] are based on segmentation that truncate carry propagation. Further, the probabilistic error analysis of these segment based adders is presented in [20, 21]. To increase the applicability of approximate designs, various accuracy configurable architectures are also presented which are reviewed in the next subsection.
2.3. Accuracy configurable architectures

In the direction of variable accuracy design, most of the architectures employ approximate adder to compute sum and with EDC logic to achieve golden output whenever required. The generic accuracy configurable adder (GeAr) [22] changes the inputs to the sub-adders and re-evaluates the partial sums to achieve output with higher accuracy. Further, five variable latency speculative adder (VLA) architectures based on high performance parallel prefix adder are presented in [23]. The VLA reduces large errors using simple EDC logic which occurred in 2’s complement addition. A new carry maskable FA (CMFA) is presented in [24] which is then utilized at few LSBs to design a carry maskable adder (CMA). However the CMA reduces dynamic power consumption over the conventional FA, it delivers very high error rate. An iterative accuracy programmable adder is proposed in [25] which can reconfigure the probability of getting correct output. A high speed CSLA architecture that provides improved power-delay product (PDP) using Manchester carry chain is presented in [26].

Recently, a simple accuracy reconfigurable adder (SARA) is presented in [27] that divides the operands into multiple smaller-sized segments and computes the approximate sum using predicted or accurate carry-in. In SARA, the predicted carry-in is generated by previous one bit and therefore provides large error rate when operated in approximate mode while exhibits poor performance over the RCA when operated in accurate mode. Finally, a quality scalable non-zeroing bit-truncation (NzBt) based adder is presented in [13] where additional control logic is added at few LSBs to force input operands into complementary values. However the NzBt adder reduces the power consumption when operated in approximate mode, it requires significantly large area and consumes more power over the exact RCA when operated in the accurate mode. Recently, a RAP-CLA [12] is presented in which longest carry propagation path is truncated when operated in approximate mode. However, the complexity of RAP-CLA is large which makes it power and area inefficient.

From the literature review, it is observed that RCA based approximate adders show high power and area efficiency at the cost of large delay. On the other hand, CLA based approximate adders provide higher performance with small area/power penalty. The approximate adders based on the CSK technique provide moderate design parameters between RCA and CLA. Therefore, to achieve high performance design, in this paper, CSK based adders are reviewed in detail and novel architecture is presented.

3. Analysis of state-of-the-art carry skip adders

From the literature review, it can be seen that the adders presented in [6] and [10] are the best available CSK scheme based accurate and approximate adders, respectively. The detailed analysis of the logic operations involved in these adders is required for the identification of redundant logics to reduce the implementation complexity. These adders are built by cascading the small size adders (called segments) of identical or variable sizes. Therefore, an $m$-bit adder segment is considered for the analysis. This section presents an analysis of the accurate adder segment followed by the approximate adder segment.

3.1. Analysis of accurate adder segment

An $m$-bit $j^{th}$ adder segment of the CSK adder presented in [6] is shown in Figure 1. It consists of an $m$-bit RCA, an incrementation block and an output carry generation logic. The RCA computes intermediate sum $(s^0 = \{s^0_{m-1}, s^0_{m-2}, \ldots, s^0_0\})$, and carry-out ($c^0_{out}$) signals using $m$-bit inputs ($x$ and $y$). An incrementation block computes the final sum-bits $(s = \{s_{m-1}, s_{m-2}, \ldots, s_0\})$ from the intermediate sum signals and input carry $c_{in}^0$. The final output carry signal ($c_{in}^{j+1}$) is generated in complement form using AND and AOI gates.
The logic operations performed in this adder segment are as follows:

**Logic operations performed in RCA:**

\[ p_i = x_i \oplus y_i; \quad g_i = x_i \cdot y_i \]  \hspace{1cm} (1a)
\[ c_i = g_i + p_i \cdot c_{i-1}; \quad c_{-1} = 0 \]  \hspace{1cm} (1b)
\[ c_{i}^{out} = c_{m-1} \]  \hspace{1cm} (1c)
\[ s_{i}^{0} = p_i \oplus c_{i-1} \]  \hspace{1cm} (1d)

**Output carry generation operation:**

\[ c_{i}^{j+1} = c_{i}^{out} + c_{i}^{j} \cdot \prod_{i=0}^{m-1} s_{i}^{0} \]  \hspace{1cm} (1e)

**Logic operations performed in the incrementation block:**

\[ z_i = s_{i}^{0} \cdot z_{i-1}; \quad z_{-1} = c_{i}^{j} \]  \hspace{1cm} (1f)
\[ s_i = s_i^{0} \oplus z_{i-1} \]  \hspace{1cm} (1g)

where \( 0 \leq i \leq m - 1 \). From above equations, we observed that:

i. The computation of \( i^{th} \) sum bit requires three XOR operations (First computes propagate \( p_i \), second computes intermediate sum \( s_{i}^{0} \) while the third computes final sum \( s_i \)).

ii. Similarly, the output carry requires two XOR operations (First computes propagate \( p_i \) whereas second computes intermediate sum \( s_{i}^{0} \)) and AND-OR-Invert operation.

If we use propagate signal to generate sum and carry bits in place of intermediate sum, it will save \( m \)-XOR operations in \( m \)-bit adder segment. Therefore, the existing adder [6] exhibits redundant operations which can be eliminated to achieve adder segment with low-logic complexity. The next subsection presents an analysis of existing approximate adder segment.
3.2. Analysis of approximate adder segment

An \( m \)-bit \( j^{th} \) adder segment [10] is shown in Figure 2. It consists of a sub-carry generator (SCG), multiplexer selection logic (MSL), multiplexer (MUX) and an \( m \)-bit sub-adder (SA). The SCG produces output carry \( (c^j_{\text{out}}) \) and a group-propagate \( (P^j) \) signal. The MSL generates select signals for the MUX to select the desired carry-out using group propagate signals \( (P^j, P^{j-1}, \ldots) \) generated by SCGs. The sub-adder produces sum-bits using \( m \)-bit inputs \( (x, y) \) and carry-in \( (c^j_{\text{in}}) \) which comes from the previous segment. The logic operations performed in the \( j^{th} \) SCG and sub-adder are given by (2).

**Logic operations performed in SCG:**

\[
p_i = x_i \oplus y_i; \quad g_i = x_i \cdot y_i \quad (2a)
\]

\[
P^j = p_{i-1} \cdot p_{i-2} \cdots p_0 \quad (2b)
\]

\[
c^j_{\text{out}} = g_{i-1} + p_{i-1} \cdot g_{i-2} + \cdots + p_{i-1} \cdots p_1 \cdot g_0 \quad (2c)
\]

**Logic operations performed in SA:**

\[
p_i = x_i \oplus y_i; \quad g_i = x_i \cdot y_i \quad (2d)
\]

\[
c_i = g_i + p_i \cdot c_{i-1}; \quad c_{-1} = c^j_{\text{in}} \quad (2e)
\]

\[
s_i = p_i \oplus c_{i-1} \quad (2f)
\]

where \( 0 \leq i \leq m - 1 \). It can be observed that the (2a) and (2d) are identical. Therefore, one set of propagate and generate operations can be removed and other set can be shared between SCG and SA which save \( m \)-XOR and \( m \)-AND operations. The MSL circuit and carry selection MUX are different for the different values of \( v \) (representing number of carry-out signals generated by SCGs used for carry prediction). However the micro-architecture of the MSL and MUX are not given in [10], the implementation corresponds to the given
functionality is shown in Figure 3 for \( v = 2, 3 \) and 4. From Figure 3, it can be seen that the logic circuit of MSL is different for different value of \( v \) i.e. there is no regularity. If the MSL along with carry selection MUX are designed with the regular structure, the resultant design would eliminate the irregularity and would provide a low complexity design.

![Truth Table and Boolean Function (\( v=2 \))](image)

![Truth Table and Boolean Function (\( v=3 \))](image)

![Truth Table and Boolean Function (\( v=4 \))](image)

\[ S = P^j \]

![Figure 3. MSL circuit and carry selection multiplexer of approximate adder segment [10] with truth-table for \( v = 2, 3 \) and 4.](image)

4. Proposed low-complexity approximate adder segment

The proposed low-complexity approximate carry-skip adder segment shown in Figure 4 is derived on the basis of analysis presented in the previous section. It consists of sub-carry generation cum propagate-generate (PG) unit, final sum-generation (FSG) unit and a new carry prediction logic (CPL) called New-CPL. The SCG cum PG unit

![Figure 4. Proposed approximate adder segment with new carry prediction logic.](image)
computes the propagate \((p = \{p_{m-1}, p_{m-2}, \cdots, p_0\})\) and generate \((g = \{g_{m-1}, g_{m-2}, \cdots, g_0\})\), group-propagate \((P^j)\) and carry-out \((c^j_{\text{out}})\) signals from the \(m\)-bit inputs \((x\) and \(y)\). The final sum \((s = \{s_{m-1}, s_{m-2}, \cdots, s_0\})\) is produced by FSG unit using \(p\), \(g\) and \(c^j_{\text{in}}\) signals. The SCG cum PG unit performs the following logic operations:

**Operations performed in SCG cum PG unit:**

\[
\begin{align*}
p_i &= x_i \oplus y_i; \quad g_i = x_i \cdot y_i \tag{3a} \\
P^j &= \prod_{i=0}^{m-1} p_i \tag{3b} \\
c^j_{\text{out}} &= g_{i-1} + p_{i-1}g_{i-2} + \cdots + p_1g_0 \tag{3c}
\end{align*}
\]

where \(0 \leq i \leq m - 1\). The operation of New-CPL is described in the truth-tables shown in Figure 5 for the different values of \(v\). The logic circuits are developed for New-CPLs according to their operations (given in truth-table) using 2-to-1 multiplexer(s) as shown in Figure 5 for \(v = 2, 3\) and \(4\). Further, it is observed that if group propagate output bit \((P^j)\) of SCG-cum-PG unit is set to logic ‘1’, its carry-out bit \((c^j_{\text{out}})\) will always be logic ‘0’, i.e., if \(P^j = 1\) then \(c^j_{\text{out}} = 0\). Therefore, design of CPL by exploiting this property requires only

**Figure 5.** Truth tables and logic circuits of New CPL for \(v = 2, 3\) and \(4\).
one AND and one OR gates in place of 2-to-1 multiplexer which provides the saving of one AND and one NOT gates. It is clear that the proposed New-CPLs exhibit low-logic complexity with regular structure over the carry prediction logic of [10].

The logic diagram of proposed adder segment is shown in Figure 6. This circuit is derived using Boolean expressions given by (3) and AND-OR implementation of the proposed New-CPL. It can be observed from Figure 6 that the proposed adder segment has low-logic complexity and exhibits structural regularity. However in Figure 6, the New-CPL is implemented for \( v = 4 \), it can be extended to any value of \( v \) by addition/deletion of AND-OR logic. The logic complexity analysis of the proposed adder segment is presented in the next subsection.

**Figure 6.** Logic diagram of the proposed \( j^{th} \) approximate adder segment with New-CPL \( (v = 4) \).

### 4.1. Theoretical analysis of the logic complexity

For the theoretical logic complexity analysis, we have taken area of 2-input AND, 2-input OR and NOT, 2-input XOR, and 3-input AOI gates from TCBN65GPLUS TSMC 65 \( nm \) core library databook [28]. The area of AND, OR, NOT, XOR and AOI are 6, 6, 2, 12, and 6 respectively in terms of transistor count. Based on the transistor count, the area estimation relation is given by (4).

\[
\text{Area} = 6N_a + 6N_o + 2N_i + 12N_{xor} + 6N_{aoi}
\]

where \( N_a, N_o, N_i, N_{xor} \) and \( N_{aoi} \) represents the gate-counts of AND, OR, NOT, XOR and AOI gates respectively. Using (4), the total area of the each design can be calculated by using the total gate-count in the design. The generalized gate counts of \( m \)-bit proposed and existing adder segments are given in Table 1. Using
the gate-count given in Table 1 and (4), we have calculated area of the proposed and existing adder segments for 8, 16 and 32 bits, and calculated values are presented in Table 2 for comparison. Table 2 shows that the adder segments proposed in [10] and [6] require 30.15% and 8.7% more area over the proposed adder segment. Consequently, it is expected that the approximate adder built from the proposed adder segment will occupy less area and consume less power over the existing approximate adder of [10].

Table 1. General expressions for gate-counts in m-bit CSK adder segments.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>4m - 3</td>
<td>5m - 3 + α_a</td>
<td>4m + v - 5</td>
</tr>
<tr>
<td>OR</td>
<td>m - 1</td>
<td>2m - 2 + α_o</td>
<td>2m + v - 3</td>
</tr>
<tr>
<td>NOT</td>
<td>α_i</td>
<td>α_i</td>
<td>α_i</td>
</tr>
<tr>
<td>XOR</td>
<td>3m - 1</td>
<td>3m - 1</td>
<td>2m</td>
</tr>
<tr>
<td>AOI</td>
<td>1</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

LEGEND: ACC-AS: Accurate adder segment, APP-AS: Approximate adder segment, \( v \): number of carry-out signals (with carry-in = ‘0’) generated from the segments (e.g. \( v = 1 \) indicates current segment only while \( v = 2 \) indicates current and previous one segment) used for carry prediction, \( (α_a, α_o, α_i) \): Number of AND, OR and NOT gates respectively in MSL and MUX for given value of \( v \).

Table 2. Theoretical comparison of the area for CSK adder segments

<table>
<thead>
<tr>
<th>Adder Segment</th>
<th>Bit-width (m)</th>
<th>Area (transistor count)</th>
<th>Excess Area (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACC-AS[6]</td>
<td>8</td>
<td>498</td>
<td>7.79</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>1026</td>
<td>8.91</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>2082</td>
<td>9.46</td>
</tr>
<tr>
<td>APP-AS[10]</td>
<td>8</td>
<td>602</td>
<td>30.30</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>1226</td>
<td>30.14</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>2474</td>
<td>30.07</td>
</tr>
<tr>
<td>Proposed</td>
<td>8</td>
<td>462</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>942</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>1902</td>
<td>–</td>
</tr>
</tbody>
</table>

LEGEND: ACC-AS: Accurate adder segment, APP-AS: Approximate adder segment. For area estimation \( v = 2 \) is considered.

5. Proposed PAEA-CSK adder

The generalized architecture of proposed PAEA-CSK adder shown in Figure 7 is derived from the proposed adder segment (shown in Figure 4). It receives \( n \)-bit inputs (\( A \) and \( B \)) and computes the \( n \)-bit sum \( S \) and carry (\( c_{out} \)) signal. In the proposed adder, conventional RCA sub-adder is used at first and last stages to reduce the logic complexity. The RCA-I computes \( m \)-bit sum (\( S_{m-1:0} \)) and carry-out signal (\( c_{out} \)) using lower \( m \)-bits of the inputs whereas, the RCA-II computes \( m \) MSBs of sum and output carry using \( m \) MSBs of input operands and carry-in (\( c_{in}^{m-1} \)) signal received from the previous segment. On the other hand, an intermediate \( j^{th} \) segment receives \( m \)-bit inputs (\( A_{j}^{j-1:j-1} \) and \( B_{j}^{j-1:j-1} \)), carry-in (\( c_{in}^{j} \)), carry outputs (\( c_{out}^{j-1}, c_{out}^{j-2}, \cdots \))
and group propagate signals \( (P_{j-1}^{\text{out}}, P_{j-2}^{\text{out}}, \ldots) \) from previous segments, and generates the sum \( (S_{jm-1:(j-1)m}) \), carry output \( (c_{jn}^{j+1}) \), group propagate \( (P^j) \) and output carry \( (c_{out}^j) \) signals, where \( 1 \leq j \leq \frac{n}{m} \). The synthesis results and quality analysis are presented in the next section to evaluate the efficacy of the proposed design.

### 6. Synthesis results and quality analysis

To evaluate the effectiveness, the proposed and existing designs are implemented in VHDL and synthesized using Synopsys Design Compiler (SDC) with 65nm TSMC CMOS Library. Further, the quality analysis is done by implementing them in MATLAB and simulating with one million random input patterns. Finally, the quality analysis of the proposed adder in real application is done by implementing Sobel edge detector (SED) embedded with proposed and existing adders. These SEDs are simulated with benchmark Lena image and error metrics are computed [29]. Following subsections first provide the synthesis results of adder segments followed by the synthesis results of approximate adder architectures and finally present the quality analysis of these adders as standalone unit and in the application.

#### 6.1. Synthesis results of adder segments

We have implemented the proposed and existing adder segments for different bit-width (\( m = 8, 16 \) and 32 with \( v = 2 \)) and synthesized them to achieve design metrics. The area, delay and power reported by the SDC for various adder segments are listed in Table 3. The synthesis result of Table 3 confirms the theoretical calculations given in Table 2. The proposed adder segment on average (for the different bit-width) consumes 20.97% and 13.14% less power over the adder segment presented in [6] and [10], respectively. The comparison of area-delay-product (ADP) for the proposed and existing adder segments is shown in Figure 8.

It can be observed from the Figure 8 that the proposed adder segment on average reduces ADP by 55.3% and 46.7% over the adder segment presented in [6] and [10], respectively. From the synthesis results,
Table 3. Comparison of ASIC synthesis results for CSK adder segments

<table>
<thead>
<tr>
<th>Adder Segment</th>
<th>Bit-width (m)</th>
<th>Delay (ns)</th>
<th>Area (µm²)</th>
<th>Power (µW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACC-AS [6]</td>
<td>8</td>
<td>0.37</td>
<td>360.7</td>
<td>10.97</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>0.69</td>
<td>661.3</td>
<td>22.62</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>1.35</td>
<td>1310.0</td>
<td>46.00</td>
</tr>
<tr>
<td>APP-AS [10]</td>
<td>8</td>
<td>0.35</td>
<td>326.1</td>
<td>10.06</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>0.68</td>
<td>566.2</td>
<td>20.31</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>1.29</td>
<td>1114.9</td>
<td>42.09</td>
</tr>
<tr>
<td>Proposed</td>
<td>8</td>
<td>0.33</td>
<td>206.6</td>
<td>8.84</td>
</tr>
<tr>
<td></td>
<td>16</td>
<td>0.65</td>
<td>288.0</td>
<td>17.77</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>1.26</td>
<td>588.9</td>
<td>35.87</td>
</tr>
</tbody>
</table>

LEGEND: ACC-AS: Accurate adder segment, APP-AS: Approximate adder segment. Power is estimated with normalized clock at 200 MHz.

it is clear that the proposed design is efficient in terms of ADP and power. Therefore, it is expected that the proposed approximate adder (shown in Figure 7) will provide significant power and ADP saving over the existing approximate adders.

Figure 8. Comparison area-delay products (ADPs) of adder segments for varying bit-width.

6.2. Synthesis results and quality metrics products of proposed PAEA-CSK

We have implemented the 16-bit proposed (using generalized architecture shown in Figure 7) and the existing approximate CSK adder (reported in [10]) for different set of m and v values (e.g. m = 2, v = 2; m = 4, v = 2; m = 2, v = 3), in VHDL. Also, we have implemented 16-bit RAP-CLA with window size equal to 4 [12] and NzBt adder [13] for comparative analysis. The area, delay and power as tabulated in Table 4 are obtained from synthesis using SDC. From the synthesis results, it is clear that the proposed approximate adder on average (for different set of m and v values) reduces ADP and power by 33.36% and 18.03% respectively over the existing similar adder design [10]. The proposed 16-bit adder (with m = 2, v = 3) consumes 47.03% less power and occupies 77.99% less area and also offers 79.82% less area-delay-product (ADP) over the RAP-CLA [12]. Further, the proposed 16-bit adder (with m = 2, v = 3) offers 60.0% less delay and 60.48% less ADP over the NzBt adder [13].
Table 4. Comparison of ASIC synthesis results and quality metrics for 16-bit Adders

<table>
<thead>
<tr>
<th>Adder</th>
<th>Design metrics</th>
<th>Quality metrics</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$m, v$</td>
<td>Delay</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(ns)</td>
</tr>
<tr>
<td>NzBt [13]</td>
<td>--</td>
<td>0.55</td>
</tr>
<tr>
<td>RAP-CLA [12]</td>
<td>--</td>
<td>0.24</td>
</tr>
<tr>
<td>ACSKA [10]</td>
<td>2, 2</td>
<td>0.20</td>
</tr>
<tr>
<td></td>
<td>4, 2</td>
<td>0.30</td>
</tr>
<tr>
<td></td>
<td>2, 3</td>
<td>0.26</td>
</tr>
<tr>
<td>Proposed</td>
<td>2, 2</td>
<td>0.19</td>
</tr>
<tr>
<td></td>
<td>4, 2</td>
<td>0.28</td>
</tr>
<tr>
<td></td>
<td>2, 3</td>
<td>0.22</td>
</tr>
</tbody>
</table>

ADP: area delay product, $m$: bit-width of the segment, $v$: number of carry-out signals generated from the segments (e.g. $v = 1$ indicates current segment only while $v = 2$ indicates current and previous one segment) used for output carry prediction. Power is estimated with normalized clock at 200 MHz; MED: mean error distance, NED: Normalized error distance.

For the assessment of the adder output quality, we have evaluated the mean error distance (MED), normalized error distance (NED) and error rate (ER) of the proposed and existing approximate adders, and summarized in Table 4. Quality result shows that the proposed adder, on average, for the different set of design parameters ($m, v$) provides 56.13%, 55.41% and 67.43% less MED, NED and ER respectively over the RAP-CLA [12]. Compared to NzBt [13], the proposed adder (with $m = 4$ and $v = 2$) involves 97.49% less NED and 96.98% less ER. Since the proposed architecture is implemented with reduced complexity without changing functionality of [10], the quality metrics of the proposed adders are same as [10]. Finally, for the quality analysis in the application, SED embedded with proposed and existing adders are implemented and simulated with Lena image. Since, the quality metrics of the proposed and [10] are same, only images processed by the SEDs with proposed adders are shown in Figure 9. Further, the SED with proposed adder provides a minimum of 16.94 dB higher PSNR over the SED embedded with NzBt adder [13]. From the Figure 9, it can be observed that proposed adders provide acceptable image quality with higher power efficiency over the RAP-CLA [12].

![Figure 9](image_url)

Figure 9. Lena image (256x256): a) original image, edge detected by Sobel edge detector embedded with b) accurate c) NzBt, d) RAP-CLA, and proposed adders with e) $m = 2, v = 2$ f) $m = 2, v = 3$, and g) $m = 4, v = 2$. 
7. Conclusion

In this paper, various logic operations performed in the existing adder architectures are analysed and redundant operations are identified. Further, a novel approximate adder segment architecture is presented by eliminating redundant logic and sharing common logics. The synthesis results show that the proposed adder segment provides on average 13.14% less power and 46.7% less ADP than the ACSKA segment. Using proposed adder segment, a new approximate adder is proposed that reduces implementation complexity and power over the existing approximate adders. The 16-bit proposed approximate adder found best in terms of power consumption and ADP than the best available approximate adder. Finally, the proposed approximate adder on average (for different set of $m$ and $v$ values) reduces 33.36% and 18.03% ADP and power respectively over the existing ACSKA.

References


[28] “dbtcbn65gplusbc0d88 TSMC 65nm CMOS library databook.”