xUAVs: Towards Efficient Approximate Computing for UAVs—Low Power Approximate Adders With Single LUT Delay for FPGA-Based Aerial Imaging Optimization

TUHA NOMANI1,2, MUJAHID MOHSIN1, ZAHID PERVAIZ1, AND MUHAMMAD SHAFIQUE3, (Senior Member, IEEE)

1College of Aeronautical Engineering (CAE), National University of Sciences and Technology (NUST), Islamabad 44000, Pakistan
2Centre of Artificial Intelligence and Computing (CENTAIC), Rawalpindi 46000, Pakistan
3Institute of Computer Engineering, Technische Universität Wien (TU Wien), 1040 Vienna, Austria

Corresponding author: Mujahid Mohsin (mujahid.mohsin@cae.nust.edu.pk)

ABSTRACT High Definition (HD) image processing and real-time analytics over live video feeds have always been the key requirements for Intelligence, Surveillance and Reconnaissance (ISR) applications. With the evolution of optics and image enhancement techniques, computational loads of HD ISR systems are also rising exponentially. On the contrary, the slow-down of Moore’s Law has recently posed challenging bounds over the level of achievable miniaturization for emerging processing and storage units. Field Programmable Gate Arrays (FPGAs) offer a popular choice of implementing ISR algorithms over resource-constrained platforms, such as Unmanned Aerial Vehicles (UAVs), due to favorable features of reconfigurability and rapid prototyping. A promising solution to bridge the gap between resource-constrained host platforms and computation-intensive FPGA applications is the paradigm of Approximate Computing. It compromises on the accuracy of processed results to offer significant performance gains for error-tolerant applications, such as video and image processing. In this paper, we present a novel approximate adder design methodology, for FPGA-based systems with improved SWaP performance, besides preserving the accuracy requirements within acceptable thresholds. The design methodology proposed in this paper focuses on the FPGA-specific Look-Up Table (LUT) architecture to introduce approximations while splitting the carry chain into LUT-based sub-adders, with flexible overlap to tune the adder’s accuracy and achieve the overall latency of a single LUT. The paper presents several variants of the proposed design and offers application-oriented flexibility to adjust for optimal SWaP vs accuracy trade-off. We have further devised a comprehensive assessment approach to verify functional viability of the proposed atomic arithmetic blocks at system level, through their implementation into dense computational imaging applications, such as 2-dimensional Discrete Cosine Transform (DCT), airborne self-localization and moving object tracking algorithms, in comparison with other state-of-the-art adders. Our most accurate design performs at least 9.9% better in power consumption when compared with existing approximate adders, which proves that the proposed methodology holds promising potential to improve SWaP-index for computation-intensive UAV applications.

INDEX TERMS Aerial ISR applications, approximate adders, approximate computing, FPGA, moving object tracking, self-localization.

I. INTRODUCTION

High-resolution aerial imaging and automated yet action- able video analytics are widely-being deployed by military and other government-sponsored law enforcement agencies for on-demand Intelligence, Surveillance and Reconnaissance (ISR) operations, autonomous Combat, Search and Rescue missions, hostile target containment and border area patrol.

Apart from military use, light-weight and low cost aerial imaging solutions find great utility in diverse commercial applications, including self localization, target tracking,
crowd management, drone swarms, collision avoidance and disaster relief operations. However, the first and foremost challenge for resource-starving aerial platforms, such as Unmanned/Micro Air Vehicles (UAVs/MAVs), is management of system resources to meet operational requirements within the allowable Size, Weight and Power (SWaP) constraints.

Applications like moving target detection and self-localization require real-time video processing through implementation of a series of complex algorithms, which are computation-intensive and latency-sensitive. The resource-constrained aerial platforms need to optimize the SWaP signatures while satisfying these challenging real-time processing constraints. Approximate Computing is one such emerging trend that allows to enhance SWaP-index by compromising the precision or accuracy of the computations. This accuracy trade-off allows the computational process to be further simplified to enhance power efficiency, use less hardware resources and reduce critical path delays.

### A. TARGET RESEARCH APPLICATION

The error-tolerant applications are more relevant to approximate computing based optimizations such as speech, image and video processing. Most of the work in the area of approximate computing has been proposed for Application Specific Integrated Circuits (ASICs) [1]–[5]. ASICs are designed as a set of logic gates and minimize the number of logic gates to reduce silicon area and power consumption. However, ASIC-focused optimizations cannot be directly mapped to the FPGA architecture, primarily due to the differences in their underlying design principles [6]. FPGAs comprise of Look-up Tables (LUTs) as their basic building block and therefore, require specialized LUT-specific approximation techniques. More research has been done in the area of ASIC based systems due to their vast applicability in real life applications and systems.

Nevertheless, Field Programmable Gate Arrays (FPGAs) are a preferred choice for hardware design of certain safety-critical and real-time applications (such as UAVs), due to their run-time re-configurability, rapid prototyping (less time to market) and low cost [7]. Thus, these specific platforms can also benefit from improved SWaP index performance through approximate computing. While adequate research has been done on FPGA-based approximate computing solutions [6], [8]–[10], [12], there is ample margin of improvement in the domain for more energy efficient solutions for SWaP constraint error-tolerant complex applications.

### B. NOVEL CONTRIBUTIONS

In this paper, we propose xUAV, a generic design and evaluation methodology for deployment of approximate modules within complex error-tolerant applications for FPGAs, as depicted in Fig. 1. The approach has been demonstrated by integrating novel approximate adder modules in complex video processing algorithms for resource-constrained aerial applications. The presented methodology optimizes the SWaP parameters of approximate systems to meet minimum performance requirements through a recursive process. For each iteration, the system is assessed by replacing accurate modules in the system, with suitable approximate modules. If the performance assessment stage delivers the desired results, the design is finalized with best achievable SWaP gains. Otherwise, the elements contributing towards undesirable inaccuracies are identified and subsequently optimized to enhance computational accuracy, until the desired results are achieved.

The key research contributions of this paper are summarized below:

- **Methodology for Evaluating FPGA-specific Approximate Adder Models**: We present a generic methodology for design and performance evaluation of approximate blocks for FPGA, optimized for SWaP constraint platforms, by exploiting the maximum capacity of available Look-up Tables for area optimization.

- **Novel Adder Designs**: We propose 10 variants of novel approximate arithmetic designs based on presented methodology, specifically targeting FPGA’s LUT-based architecture.

- **Mathematical Modeling Based Error Analysis**: Instead of relying on computer-based Monte-Carlo simulations, error statistics are drawn through mathematical modeling of each bit of the designed adder, as well as existing state-of-the-art adder designs for fair comparison.

- **Adder’s Evaluation for Aerial Imaging Applications**: To objectively evaluate the suitability of proposed methodology for real-world applications, the novel adder designs have been deployed in Discrete Cosine Transform function for JPEG compression, followed by quantitative analysis on self-localization and moving object tracking algorithms.

- **Open-source Model**: Verilog implementation of our novel approximate adder designs, along with their
MATLAB behavioral implementations have been made available at https://sourceforge.net/projects/fpga-approx-adder as open source, for further research in the field.

C. PAPER ORGANIZATION
The rest of the paper is organized as follows. Section II presents some of the recent research contributions in the domain of approximate computing. Section III explains the novel design methodology of LUT-based approximate adder as well as the statistical model to mathematically compute accuracies of the proposed adders. Section IV discusses the implementation results and section V concludes the paper, while giving some pointers to the planned future research efforts.

II. BACKGROUND AND RELATED WORK
This section builds the necessary background and reviews the state-of-the-art in approximate computing with regards to ASIC and FPGA-based designs.

A. ARITHMETIC APPROXIMATION
Since the development of complex DSP algorithms the arithmetic adder, being the atomic element of any image processing system, has always remained a focal-point of optimization efforts [13]. Legacy Ripple Carry Adders were evolved into faster designs of Carry-Look-Ahead Adders, Carry-Select Adders, and Carry-Save Adders etc [14]. Moreover other arithmetic operations like multiplication also instantiate multiple adder blocks. Efforts on multiplier optimizations include techniques like the Wallace and Dada reductions. For further details on these accurate arithmetic variants, readers may refer to Khan [15].

With the advent of approximation and ever rising need for SWaP optimizations, various researchers migrated from accurate to approximate computing in arithmetic modules, thus pushing the limits of achievable efficiency in processing. Some of these approaches are discussed below:

- **Precision scaling**: In this approach, minimum required precision is determined by performing profiling at the design time. At run time, the energy difference between consecutive simulation steps is measured and compared with a threshold to detect whether the simulation is stable or unstable [16].

- **Loop Perforation**: This approach performs approximation by skipping some iterations of a loop to reduce computational overheads [17].

- **Memoization**: In this strategy, results of functions are stored for later reuse. By reusing the results for similar functions/inputs, the scope of memoization can be enhanced at the cost of possible approximation [18].

- **Inexact Hardware**: The hardware designed for a specific task is optimized by compromising on results. Optimization is completely focused on enhancing SWaP parameters and reducing critical-path delays [19].

As per different surveys conducted on the scope of approximation strategies (see for example [20]) inexact hardware stands out as one of the most promising techniques for DSP applications. Its primary advantage over Memoization techniques is its generalized applicability. Memoization involves fixation of customized logic at different stages at the algorithm level. Furthermore, placement of this logic is different for different processes. While inexact hardware, is not algorithm-dependent and thus, can be implemented in a more exclusive fashion. A summary of approximation strategies adopted vis-à-vis target applications is presented in Table 1 [20].

| Technique Selected | Image Processing | Scientific Computing | Financial Analysis | Database Search |
|--------------------|------------------|----------------------|--------------------|-----------------|
| Precision Scaling  | -                | -                    | ✓                  | ✓               |
| Loop Perforation    | -                | ✓                    | ✓                  | ✓               |
| Memoization        | ✓                | -                    | ✓                  | ✓               |
| Inexact Hardware    | ✓                | ✓                    | -                  | -               |

B. ASIC-BASED INEXACT-HARDWARE MODELS
Using Inexact Hardware, numerous researchers have achieved higher performance (lesser processing delays, lower power consumption, smaller silicon footprint) and achieved good accuracy standards for approximate adders. However, most of these designs have been formulated for ASIC systems. Some of the noteworthy works are discussed below.

Reverse propagating carry chain has been suggested by Pashaeifar et al. [1], which adopts the principle of Carry Lookahead Adder and computes carry of the \( i \)th bit by using carry of \( (i + 1)th \) bit position \( (C_{i+(i+1)}) \) and its input bits. The work shows that by going in reverse, the possible solution set is approximated, thus achieving approximation from MSB towards LSB part, achieving better accuracy and stability in delay variations.

Carry-chain truncation is an effective approach adopted by several ASIC-specific proposals of approximate adders [3], [4], [21]. These efforts equated carry bit to one of the input bits to achieve simplified logic designs. While these designs exhibit moderate accuracy in comparison to their counterparts, they gain significantly on the SWaP index. Zhu et al. [2] proposed a check on input bits to implement carry-chain truncation. If the bits are zero or different, simple addition is performed. However, in case of both 1, all LSBs are computed as 1. This approach, although innovative, does not provide a uniform optimization and improves performance metrics only for specific set of inputs.

Another family of approximate adders constitutes of designs with flexible and re-configurable approximations [5], [22], [23]. These modules have integrated circuits for Error Detection and Correction (EDC) [24]. One of these designs
is Accuracy Configurable Adder (ACA-II) [5], which implies increment units with each sub-adder to correct the computed sum. In Gracefully Degrading Adder (GDA) [22], accuracy is set by turning multiplexers on and off. However, EDC circuits induce additional clock cycles and logic-gates’ overheads, diminishing the fast processing and low footprint gains of the original approximate design [11].

Recently, Jha and Mekie [25] introduced the concept of approximating sum bits instead of carry bits. Through this approach, the authors have simplified adder circuitry, thus reducing number of logic gates. Since the approximation is implemented in summation bits, the carry bits propagating towards MSBs stay accurate and thus the overall system accuracy is improved.

Using the approaches of Error Detection and Correction designs [5], [22], [23], Generic Accuracy configurable adder (GeAr) approach [11] introduces sub-adder-based designs which carry certain sub-adder output results, thus breaking the carry chain. The authors presented several variants of GeAr with GeAr (8,4) delivering the best performance in terms of accuracy. In this design, the sub-adder size of 8 bits is introduced with overlap of 4 bits among the sub-adder. However, the sub-adder based approaches as suggested by the works of [5], [11], [22], [23] are primarily aimed for the ASIC-based designs and therefore optimized accordingly. Such approaches do not scale for FPGA-based designs, as have also been demonstrated in the work of [6].

As discussed in this section, most research work in the field of Approximating Computing is ASIC-based. However, differences in underlying logic implementation in ASICs and FPGAs make these works less suitable when implemented on FPGAs.

C. FPGA-BASED APPROXIMATION MODELS

LUT focused approximation approach has been suggested by [26] and [27]. The input is first distributed into Most Significant Part (MSP) and Least Significant Part (LSP). Carry is only propagated from LSP to MSP if MSB of LSP generates a carry. In case, the carry propagation is discovered in MSB of LSP, all sum bits of LSP are set to 1, while keeping MSP additions accurate.

Prabakaran et al. [6] introduced a series of 8 variants of approximate adders (named as DeMAS) by equating input bits to carry out bit. A notable contribution by Boroumand et al. [28] proposed base 4 (Quaternary) additions. In their proposal, four bits are taken as sub-adder input to generate two carry chains i.e. $C_{i+1}$ and $C_{i+2}$ in addition to the sum. The authors suggest to proceed with only one carry chain, thus creating a Quaternary Approximate Adder.

Given the logic block differences between ASICs and FPGAs, a research gap exists to propose FPGA-specific efficient approximate adder modules. It is because ‘as is’ implementation of ASIC based designs on FPGAs do not yield similar performance gains [6]. In our proposed work, we have adopted sub-adder based carry-chain-breaking approach, with a novel effort of optimizing sub-adders through use of multiple FPGA Lookup Tables (LUTs), and thus achieving gains in critical path delays and area overheads.

III. APPROXIMATE ADDER MODEL

Our proposed approximate adder design methodology aims to extract optimum accuracy while improving SwAP-index and minimizing the critical-path delay. The overall design methodology focuses on the FPGA-specific LUT architecture and introduces approximations while splitting the carry chain into single LUT-based sub-adders, with flexible overlap to tune the overall accuracy. This section begins with a brief introduction of FPGA architecture, followed by an overview of the probabilistic model used for error calculations. It then presents the novel approximate-adder design methodology and its different adder variants, followed by a thorough analysis of respective errors induced by these adders.

A. FPGA ARCHITECTURE

FPGA differs from ASIC because of its open architecture. In ASICs, computation is realized with logic gates. However, in case of FPGAs, logic is implemented through Configurable Logic Blocks (CLBs) [29]–[31]. Typical CLB design is as shown in Fig. 2. CLB can be of two types i.e. SLICEL and SLICEM (Logic and Memory slices) and is connected to a switch matrix for access to the general routing matrix. Each slice comprises of Look-Up Tables (LUTs) and flip flops. Overall numbers of flip flops and LUTs are different for different FPGA chips. A k-input LUT is a block of SRAM, which can implement a truth table of $k$ inputs and one output. For example, 3-input LUT allows 3 bit inputs and stores $2^3$ possible single-bit outcomes, while mapping each unique input combination to deliver the correspondingly stored unique output. The latest FPGAs contain up to 6-input LUT configurations.

B. MODELING FIXED APPROXIMATION WITH SINGLE LUT DELAY

Let us consider a 2-operand adder, where the size of each operand is $X$ bits. We represent this adder as an integration of $L$ sub-adders, where each sub-adder is of size $S_o; 1 < a < X$. Each of these sub-adders has an overlap of $ov$ bits,
where $0 \leq ov < S_a$. $L$ is bounded by the following equation:

$$L = \left\lceil \frac{(X - ov)}{(S_a - ov)} \right\rceil \quad (1)$$

**Example:** Let us consider the sub-adder size ($S_a$) of 3 bits with an overlap ($ov$) of 1 bit, as shown in Fig. 3. The first (least significant) sub-adder computes 4 bit sum while taking three least-significant bits as inputs from each of the two operands. All sum bits, except carry out, is used in the resultant sum. Now, for the second sub-adder (SA-2), only the second and third output bits are considered. Please note that Sub-Adder 1 (SA-1) contributes an accurate value of $S_2$, which is therefore preferred over the erroneous output bit contributed by SA-2. Similarly, $S_4$ bit contributed by SA-2 forms part of the output sum, owing to its higher accuracy value (due to more “carry-chain-effect”), when compared with the same bit delivered by SA-3. Further sub-adder bits are utilized using the same approach. Since the carry chain for each sub-adder is truncated at LUT level, the overall approximate adder design works with a delay of 1 LUT only, as explained further in subsequent sections.

**C. 2-BIT & 3-BIT SUB-ADDER BASED DESIGNS**

As explained in section A, the design implementation in FPGAs is through LUTs. Considering sum of two operands of three bits each, the addition requires computations over six input bits to yield three output bits (ignoring the carry-in/out), which can be handled using three six-input LUTs. Each of these three LUTs shall have a six-input mapping onto a single bit output. In this way, by using three six-input LUTs, we calculate the result with a delay of single LUT. Using a maximum of 6-input LUT structure, the viable sizes of sub-adder can be either 2 or 3 bits only. Hence, the proposed sub-adder model variants with sub-adder sizes of 2 and 3 bits, implemented with single LUT delay, are depicted in Fig. 4. With the given sub-adder based design, the carry chain propagation is limited to the size of the sub-adder itself [8]. Furthermore, the total number of LUTs consumed by the design varies proportionally with the number of overlap ($ov$) bits.

In order to calculate the error probability, we have developed a specific mathematical error model instead of using Monte Carlo simulations. The approach has been verified by running exhaustive simulations for the complete sample space. Mathematical modeling enables us to successfully identify elements which are either generating or promoting error through the design, for subsequent optimization. Our model is derived from the the generalized model of the ASIC based approximate-adder modeling conducted by Mazahir et al. [32]–[34], but we specialize it for the FPGA-based designs as multi-LUT based optimizations may result in different approximate adder configurations that have not been studied in the work of [32]. Our mathematical error model defines error contributed by each sub-adder as an event $E$ e.g. error in sub-adder 3 is represented as $E_3$. Any sub-adder itself, is accurate. Error occurs in a sub-adder as carry-out from previous sub-adder is truncated. Thus the event $E$ depends upon the behavior of specific location bits. For calculating these behaviors, we name them $G_0$, $G_1$ & $P$, as defined below [32]:

- $G_0$: Set of n-bit pairs generates a carry out with carry-in = 0 and computed as $(\frac{2^n-1}{2^{n+1}})$.
- $G_1$: Set of n-bit pairs generates a carry out with carry-in = 1 and computed as $(\frac{2^n}{2^{n+1}})$.
- $P$: Set of n-bit pairs propagates a carry-in and computed as $(\frac{1}{2^n})$.

Here, $n$ is defined as the number of bits, generating or propagating a carry. The sub-adder error events $E_2, E_3, \ldots E_L$ are mutually dependent and their combined effect defines the overall error probability for $X$-bit operands through inclusion.
exclusion principle, and is given as [32]:

\[
P_r[\text{Error}] = P_r[E_2 \lor E_3 \lor \ldots \lor E_L] = \sum_{i=2}^{L} P_r[E_i] - \sum_{2 \leq i < j \leq L} P_r[E_i \land E_j] + \sum_{2 \leq i < j < k \leq L} P_r[E_i \land E_j \land E_k] - \ldots + (-1)^L P_r[\bigwedge_{i=2}^{L} E_i]
\]

(2)

Using this generic design approach, we propose the following five adder designs:

1) **Sa2ov0**
   This configuration, pictorially depicted in Fig. 4(a), keeps the sub-adder size of 2-bits without any overlap among the sub-adders. Consequently, output from the first sub-adder will be accurate; however, the error probability for second sub-adder \(E_2\) is given as:

\[
P_r[E_2] = P_r[G_{0,n=2}] = \left(\frac{2^n - 1}{2^{n+1}}\right)_{n=2} = 37.5\%
\]

(3)

2) **Sa2ov1**
   As overlap of 1-bit is introduced among sub-adders of 2-bits, the design is as seen in Fig. 4(b). Here, while computing the error term \(G_0\) is computed for carry generated by the two least significant bit-pairs (hence \(n=2\)) i.e. non-overlapping bits of SA-1 and the term P is computed for carry propagated by LSB of SA-2 (\(n=1\)). The error probability is thus given as:

\[
P_r[E_2] = P_r[G_{0,n=1} \land P_{n=1}] = \left(\frac{2^n - 1}{2^{n+1}}\right)_{n=1} \times \left(\frac{1}{2^n}\right)_{n=1} = 12.5\%
\]

(4)

3) **Sa3ov0**
   Using the sub-adder size of 3, the most basic setup of Fig. 4(c) with zero overlap bit delivers accurate values for S0, S1 and S2 sum bits. The error induced in second sub-adder due to carry truncation is represented by the following equation:

\[
P_r[E_2] = P_r[G_{0,n=3}] = \left(\frac{2^n - 1}{2^{n+1}}\right)_{n=3} = 43.75\%
\]

(5)

4) **Sa3ov1**
   As the overlap of 1 bit is introduced, as shown in Fig. 4(d), the probability of error induction \(E_2\) becomes:

\[
P_r[E_2] = P_r[G_{0,n=2} \land P_{n=1}] = \left(\frac{2^n - 1}{2^{n+1}}\right)_{n=2} \times \left(\frac{1}{2^n}\right)_{n=1} = 18.75\%
\]

(6)

5) **Sa3ov2**
   In this design, the overlap is increased to 2 bits, as represented in Fig. 4(e). The probability of error for second sub-adder for this design is:

\[
P_r[E_2] = P_r[G_{0,n=1} \land P_{n=2}] = \left(\frac{2^n - 1}{2^{n+1}}\right)_{n=1} \times \left(\frac{1}{2^n}\right)_{n=2} = 6.25\%
\]

(7)

Note that the error probabilities calculated in equations 3 - 7 are for second sub-adder due to truncation of carry out by the first sub-adder. The probability of error for the complete approximate adder has been calculated in an example discussed at the end of the following subsection. The delay of above mentioned designs remains as single LUT, independent of sub-adder size and overlapping bits.

### D. 5-BIT SUB-ADDER BASED DESIGNS

Modern FPGAs (such as Virtex-7) offer 5-LUT combinatorial blocks, which can function as 6-input LUTs with common input, as displayed in Fig. 5(a). Thus, the logic implemented within the two 5-input LUTs can be tuned in such a way that one LUT is configured for possible carry-in as ‘1’, while the other calculates result if the carry is ‘0’, as presented in Fig. 5(b). This LUT-specific sub-adder design seeks its motivation from the accurate Carry Save Adder [15]. Using this configuration, the size of this subadder is increased to 5 bits with a latency overhead of 1 built-in FPGA multiplexer. This small delay penalty comes with the advantage of larger sub-adder size, which can process a given data stream with lesser number of sub adders, as can be seen from equation 1. Furthermore, maximum resources are utilized by engaging 10 out of 11 LUT inputs. Using this approach, we propose five variants of approximate adder designs with 5 bit sub-adders, as demonstrated in Fig. 6.

For the sake of completeness and better understanding, we compute event \(E_2\) for each sub-adder variant and then present the total error contributed by our Sa5ov4 design by following the methodology presented in [32]. Here, \(E_2\) is defined as the event of error in second lower-order sub-adder. \(E_2\) occurs when the non-overlapping bit-pairs of SA-1 generate a carry and it gets propagated through the overlapping bits of SA-2.

1) **Sa5ov0**
   As shown in Fig. 6(a), the sub-adder size of 5-bits is implemented without any overlap among the sub-adders. In this configuration, S0 through S4 are computed accurately by the first sub-adder. However, carry truncation results in following error in second sub-adder:

\[
P_r[E_2] = P_r[G_{0,n=5}] = \left(\frac{2^n - 1}{2^{n+1}}\right)_{n=5} = 48.44\%
\]

(8)

2) **Sa5ov1**
   Fig. 6(b) shows the approximate design having 1-bit overlap between its sub-adders. The error event \(E_2\) for this
configuration changes as:

\[ P_r[E_2] = P_r[G_{0,n=4} \land P_{n=1}] \]
\[ = \left( \frac{2^n - 1}{2^{n+1}} \right)_{n=4} \times \left( \frac{1}{2^n} \right)_{n=1} = 23.44\% \quad (9) \]

3) Sa5ov2

Increasing sub-adder overlap to 2 bits, presented in Fig. 6(c), probability of E2 is given as:

\[ P_r[E_2] = P_r[G_{0,n=3} \land P_{n=2}] \]
\[ = \left( \frac{2^n - 1}{2^{n+1}} \right)_{n=3} \times \left( \frac{1}{2^n} \right)_{n=2} = 10.98\% \quad (10) \]

4) Sa5ov3

As the overlap is further increased to 3 bits, sum bit S5 computation is further strengthened with a longer carry-chain, thus reducing the error probability of E2 as:

\[ P_r[E_2] = P_r[G_{0,n=2} \land P_{n=3}] \]
\[ = \left( \frac{2^n - 1}{2^{n+1}} \right)_{n=2} \times \left( \frac{1}{2^n} \right)_{n=3} = 4.68\% \quad (11) \]

5) Sa5ov4

In this design, the overlap is maximized to 4 bits, giving maximum advantage of carry effect to S5 bit. As in previous 5-bit adder designs, S0 to S4 remain accurate, while E2 probability becomes:

\[ P_r[E_2] = P_r[G_{0,n=1} \land P_{n=4}] \]
\[ = \left( \frac{2^n - 1}{2^{n+1}} \right)_{n=1} \times \left( \frac{1}{2^n} \right)_{n=4} = 1.56\% \quad (12) \]

Example: In equations 8 - 12, the probability of event (E2) is computed independently for each sub-adder variant. Calculation of error probability for a complete adder module depends on its length X, which can be simplified using the methodology presented in [32]. In this example, we compute the error for design Sa5ov4 having a size (X) of 8 bits.
Using Eq 2, the error probability of this design is given as:

\[ P_r[E_{sub_{6\text{4}}}] \mid X = 8 \]
\[ = P_r[E_2 \vee E_3 \vee E_4] \]
\[ = \sum_{i=2}^{4} P_r[E_i] - \sum_{2 \leq i < j \leq 4} P_r[E_i \wedge E_j] + P_r[E_2 \wedge E_3 \wedge E_4] \]

(13)

Let us now calculate each of these terms separately. First, we calculate the joint probability of individual events \( E_2, E_3 \) & \( E_4 \):

\[ \sum_{i=2}^{4} P_r[E_i] \]
\[ = P_r[E_2] + P_r[E_3] + P_r[E_4] \]
\[ = P[G_{0,n=1} \wedge P_n=4] + P[G_{0,n=2} \wedge P_n=4] \]
\[ + P[G_{0,n=3} \wedge P_n=4] \]
\[ = \left[ \left( \frac{2^n - 1}{2^{n+1}} \right)_{n=1} \times \left( \frac{1}{2^n} \right)_{n=4} \right] + \left[ \left( \frac{2^n - 1}{2^{n+1}} \right)_{n=2} \times \left( \frac{1}{2^n} \right)_{n=4} \right] \]
\[ + \left[ \left( \frac{2^n - 1}{2^{n+1}} \right)_{n=3} \times \left( \frac{1}{2^n} \right)_{n=4} \right] = \frac{17}{256} \]

(14)

For the second term of equation 13:

\[ \sum_{2 \leq i < j \leq 4} P_r[E_i \wedge E_j] \]
\[ = P_r[E_2 \wedge E_3] + P_r[E_2 \wedge E_4] + P_r[E_3 \wedge E_4] \]
\[ = P[G_{0,n=1} \wedge P_n=5] + P[G_{0,n=1} \wedge P_n=6] \]
\[ + P[G_{0,n=2} \wedge P_n=5] \]
\[ = \left[ \left( \frac{2^n - 1}{2^{n+1}} \right)_{n=1} \times \left( \frac{1}{2^n} \right)_{n=5} \right] + \left[ \left( \frac{2^n - 1}{2^{n+1}} \right)_{n=1} \times \left( \frac{1}{2^n} \right)_{n=6} \right] \]
\[ + \left[ \left( \frac{2^n - 1}{2^{n+1}} \right)_{n=2} \times \left( \frac{1}{2^n} \right)_{n=5} \right] = \frac{3}{128} \]

(15)

The third term can be computed as:

\[ P_r[E_2 \wedge E_3 \wedge E_4] = P_r[G_{0,n=1} \wedge P_n=6] \]
\[ = \left( \frac{2^n - 1}{2^{n+1}} \right)_{n=1} \times \left( \frac{1}{2^n} \right)_{n=6} = \frac{1}{256} \]

(16)

Substituting values from equations 14, 15 & 16 in Eq 13:

\[ P_r[Error] \mid X = 8 = P_r[E_2 \vee E_3 \vee E_4] \]
\[ = \frac{17}{256} - \frac{3}{128} + \frac{1}{256} = 4.68\% \]

(17)

Following these steps, probability of error for any of the proposed designs can be calculated for a given operand size using Eq 2. The reference equation can also be used to precisely quantify the error induced by each approximate component for subsequent optimization.

**E. HYBRID ADDER DESIGNS**

We define hybrid adder as an approximate design, which consumes variable-sized sub-adders and/or dissimilar overlapping bits, with an aim to fine-tune SWaP and accuracy requirements. During the iterative optimization process, proposed by the xUAV methodology, any combination of the proposed sub-adder models and overlaps can be selected to construct a hybrid adder, suiting a given application-specific requirement. As an example, for SWaP-critical applications, requiring higher accuracies in terms of error-magnitudes, sub-adders with higher number of overlapping bits (hence increased accuracy) can be used for higher order bits as compared to the lower order bits. Such a hybrid design will increase the computational accuracy of most significant part of the output sum, thus reducing the overall error magnitude. Besides, fewer overlapping bits in the least significant part of the adder can be used to keep SWaP-index within the desirable limits.

**IV. IMPLEMENTATION AND EVALUATION**

This section initially covers a brief explanation of the methodology adopted for implementation and evaluation of the proposed adders of xUAV-family as well as the simulation setup. Additionally, the section also presents the experimental results and their critical analysis.

**A. DESIGN & ASSESSMENT METHODOLOGY**

We evaluate the performance of our novel adder designs while following the generic methodology presented in section I (Fig. 1). The overall design and evaluation flow is demonstrated in Fig. 7. The evaluation targets deployment of our novel adders within computationally-intensive airborne video surveillance applications, including moving object tracking and self-localization algorithms. With the recent widespread adoption of lightweight unmanned aerial vehicles, the demand, and consequently the challenges, of deploying autonomous surveillance and tracking algorithms over these resource-starving platforms has also increased manifold [35], thus justifying our selection of these algorithms for evaluating performance gains through approximations. Our evaluation constitutes a multi-phase iterative process, involving the selection of an appropriate adder module, out of the proposed designs, and analyzing its suitability for the targeted application through quantification of performance parameters, based on well-defined SWaP and accuracy metrics.

Let us now look at the methodology of Fig. 7 in detail. To start the process, an initial design is made, followed by its LUT-based optimization. In this optimization, the design is tailored to utilize minimum number of target-LUT for best utilization of resources. After this step, we reach ‘proposed design’ stage, which is then modeled in three different environments, as a function of elementary sub-adder blocks: (a) **HDL model** is made in Verilog and synthesized in Vivado / Xilinx ISE. The model is synthesized independently for three different configurations, while selecting the built-in ISE features of “Area”, “Power” and “Balanced” optimization settings. Each HDL-model variant is evaluated for latency and critical-path delays. In parallel, the HDL code is also scrutinized through X-power analyzer and Vivado post-implementation power reports for on-chip
power consumptions. (b) **Mathematical model** is developed for probabilistic error computations to calculate the overall accuracy. This development is done on the error-event based calculation already discussed in Section III Subsection B.

(c) Finally, the **MATLAB model** of the adder is developed for overall performance and error analysis. This MATLAB model is initially utilized to perform error computations for all possible combinations of inputs. Alternatively, error approximation can also be done by limited random inputs if the designers are constrained on computational resources. The results achieved from the three models are correlated to analyze (and subsequently optimize) the performance of the selected adder design, before moving over to implementation at the system level.

In the system level performance phase, a system model is developed in MATLAB while leveraging the modular approximate adder design as its building block. This modular approach extends the option to flexibly replace the accurate adder modules with the approximate modules deployed within a given system-level implementation. Using this approach, our implementation modifies an accurate implementation of Discrete Cosine Transform (DCT) function into an approximate DCT by replacing all of its adder modules with approximate adders. The resulting approximate DCT was further deployed in JPEG compression of a series of frames extracted from airborne video samples, which were subsequently evaluated for automated surveillance applications, including moving object tracking and self-localization/terrain tracking algorithms. Performance assessment was quantified, based on well-defined metrics such as the number of contrast points marked and correct as well as incorrect vectors computed by the algorithms. If the threshold performance parameters were met, the design with desirable SWaP parameters was finalized. In case of below-par performance, the resulting algorithm was analyzed for identification of catalyst elements causing that shortfall and were optimized accordingly.

As an example, consider that increasing the sub-adder overlaps increases net-delays in the FPGA post-PAR analysis. In order to reduce the net delays, various combinations of overlap and their relationship was understood. Based on this understanding, the designs were refined by reducing overlap and repeating the cycle again. In parallel, reducing the overlap increases the error probability in the overall function. Therefore, through an iterative process, a tradeoff point delivering acceptable results with maximum performance gain was achieved. In the recursive optimization process, a design could either be fully optimized or declared unfit, as per the results obtained after multiple iterations. For achieving credible results, the entire process was repeated on ten different video tracking scenarios and results were compiled (presented later in this section) for different xUAV variants, alongside state-of-the-art approximate adder designs.

**B. SIMULATION SETUP**

Design verification was conducted in three phases, as presented in Fig. 7. Area and critical-path delay analysis was performed in Xilinx ISE 14.7 through on-board synthesis tools. For power calculations, Xpower Analyzer was utilized and the results were subsequently verified through Vivado 2019 post-implementation reports. Xilinx Virtex-7 was selected as the target platform for implementations. The MATLAB model was created in MATLAB2018b and all simulations were performed on Windows 10 based system with Intel Core i7-7700k CPU and 64GB RAM.

**C. RESULTS**

We have used a 12-bit adder model in our experiments to facilitate the option of exhaustive simulations as well as easy reproducibility of results by other researchers [32]. The
overall performance of proposed family of adders plotted vis-à-vis Critical Path Delay, Error Probability, Consumed Power and Area is presented in Fig. 8. Detailed analysis on each of these aspects is given below:

1) DELAY
The maximum time of any process can be measured from its critical path. Computationally, it is the worst-case timing requirements for the overall process. Considering the single LUT designs, xUAV’s Sa3ov2 configuration is 28.2% and 27.6% faster than GeAr [11] and ACA [5] approximate adders, respectively. Our design is 15% slower than DeMAS [6]; however, the DeMAS adder is heavy on the error index, as discussed later. In case of sub-adder 5 variants, the delays are approximately the same as GeAr and ACA.

2) AREA
An important resource in FPGA implementations is the design area. Larger area corresponds to higher consumed power; however, an area overhead can be used as a trade-off to improve computational accuracy or reduce critical path delays. Area evaluation reveals that xUAV’s Sa3ov2 design is comparable with implementation area of DeMAS D2 Adder [6] and performs better than state-of-the-art designs such as GeAr and ACA by 68% and 56%, respectively. With an increased sub-adder size and accuracy, model Sa5ov4 leads GeAr with an area improvement factor of 53.8%.

3) POWER
The consumed power index for the proposed Sa5ov4 design exhibits 9.9% and 10.6% savings, when compared with the state-of-the-art approximate adder models with comparable error performance, i.e. with GeAr and ACA, respectively. Although DeMAS adders exhibit improved energy savings, their moderate accuracy limitation render them unsuitable for video surveillance applications, as discussed later.

4) ACCURACY
Instead of relying on Monte-Carlo simulations, we have used the actual mathematical model of the approximations to compute error probabilities. Additionally, the calculations have been verified with the help of MATLAB-based simulations of 12-bit adders for the complete sample space. The errors contributed by different designs are summarized in Table 2. Among the proposed designs, Sa3ov2, Sa5ov3 and Sa5ov4 have the highest overlapping areas (two bits or more). Consequently, these designs perform better on accuracy index, when compared with other variants. While the error probability of Sa3ov2 remains high (52%) as compared to Sa5ov3 (18.21%) and Sa5ov4 (9.3%), Sa3ov2 offers lesser implementation area and critical path delay. Therefore, if an application is tolerant enough to function with the error probability of Sa3ov2 model, this design can offer noteworthy performance gains as comparison to more accurate xUAV variants.

For comprehensive performance evaluation, we have also computed the Mean Error Distance (MED) [36] and Power-Delay-MED-Product (PDMP) [37] values for different xUAV variants and the competing adders, as given in Table 2. It can be deduced from the table that as the sub-adder overlap is
increased for xUAV variants, the MED decreases. While the increase in the overlap bits also add slight power and delay overheads, the net effect is reduction in the PDMP value due to significant improvement in the MED.

D. DISCUSSION

For detailed comparative analysis, simulation results of competing parameters for under-discussion approximate adders are tabulated as Table 2 and the same are also presented as a radar graph in Fig. 9. The radar graph truly depicts the inherent trade-offs among these parameters, thus further supporting the case that a perfect adder, which outperforms in all performance metrics, cannot be realized. The essence here is to find the best possible combination of under consideration parameters, which can optimally meet the application-specific requirements (also referred as minimum acceptable accuracy [38]), thus attaining a perfect balance among SWaP, processing delays and accuracy for real-time and computationally intensive applications. Here, by the term 'application-specific requirements', we refer to the overall aim of the targeted system being optimized with approximate adders. As an example, we may consider a moving object tracking algorithm, which should be able to identify moving targets in consecutive images of a video with at least 90 percent confidence. This defines our ‘application-specific requirement’ which is expected out of the system. The goal now, is to optimize SWaP parameters while conforming to the laid down system requirements. In real life scenarios, system can be composed of any error-tolerant application and designed adder model, when substituted in the system, should be able to keep the system operating within the acceptability limits, as defined earlier.

E. APPLICATION OF DCT FOR AERIAL IMAGERY

As a case study, following our proposed methodology in Fig. 7, we integrated our adder models in Discrete Cosine Transform (DCT), which was further deployed inside JPEG compression algorithm. This was achieved by first implementing the MATLAB models of our proposed adders as well as some of the state-of-the-art designs. Subsequently, approximate adders were integrated inside a DCT compression function using MATLAB2018b. Image compression results of accurate DCT, alongside those produced by different approximate designs, are shown in Fig. 10.

The figure shows a graceful degradation of compressed image quality with reduction in the accuracy of deployed adders. Image qualities of GeAr and xUAV-Sa5ov4, having same Peak Signal to Noise Ratio (PSNR) values, closely match with the accurate DCT-based JPEG compression output (Fig. 10 (b)), owing to their negligible error-index. However, as discussed earlier, Sa5ov4 design delivers better SWaP ratings when compared with GeAr and ACA. The image quality achieved by Sa5ov4 holds the basic integrity of the processed image with at least

![FIGURE 9. Radar graph representation of proposed design vis-a-vis state-of-the-art approx adders (12-bits).](image-url)

![FIGURE 10. Processing results with introduction of different approximate adder models in Discrete Cosine Transform function.](image-url)
27.6% faster execution and 56% less-area implementation. Contrarily, low-accuracy adder models like ACA-II, xUAV-Sa2ov1 and xUAV-Sa3ov2 exhibit significant visual degradations in image quality, as evident from loss in their respective PSNR values, given in Fig. 10.

F. SELF ESTIMATION THROUGH TERRAIN TRACKING

A number of researchers have utilized aerial imagery and video coding for evaluation of hardware accelerator modules [39], [40]. Following the approach presented in Fig. 7, we conducted a comprehensive quantitative assessment of xUAV and competing adder models after deploying them in self-localization algorithm. Self-localization techniques are widely-deployed in modern-day aerial platforms for enhanced situational awareness and augmenting precise position calculations from Global Positioning System (GPS) and/or Inertial Navigation Systems (INS). It leverages real-time video feeds from on-board sensors for location correction of GPS/INS and can ensure navigation in GPS denied environments or in case of INS failures. In order to estimate/update self-location, aerial platforms compare consecutive frames of on-board camera video and mark points of high contrast ratio. Subsequently, same ratio marks under multiple consecutive frames are joined, thus forming a vector direction of terrain movement, which actually corresponds to movement direction of the aerial platform with respect to visual terrain.

In order to gauge the performance of self-localization for different approximate adders, multiple image-pairs were extracted from a drone’s on-board camera. These pairs were then passed through an approximate self-localization algorithm to automatically generate terrain vectors by adjoining similar points. The algorithm preprocesses the input images through approximate DCT, detects its Harris features and resultanty draws terrain vectors. The results of this algorithm over a sample frame-pair are presented in Fig. 11. For each approximate adder under test, the image displays terrain vectors as lines drawn between similar marks on both frames, represented by red circles (Frame-1) and yellow crosses (Frame-2). Taking the accurate model (Fig. 11 (a)) as the reference, it can be observed that these vector lengths are more for features close to the drone as compared to the distant features and therefore, both the magnitude and direction of these terrain vectors are processed to estimate the location of the drone.

The performance of approximate modules was evaluated with the help of two benchmarks, with reference to the accurate model; firstly the number of markers/features acquired from an image-pair, and secondly, the accuracy of obtained vectors. Comparative results of this analysis is presented as Fig. 12. It can be observed that the number of marked points and resulting count of correct and incorrect vectors vary significantly for approximate algorithms and directly conforms to the probability of error induced by the corresponding approximate adder, deployed in that system. Choice of a suitable approximate algorithm for self-localization application depends upon the user-defined threshold of false-positives and false-negatives and therefore, an optimal design can be short-listed with the help of an iterative process.
G. MOVING OBJECT TRACKING

Another very commonly used application for light-weight aerial vehicles is autonomous tracking of moving objects. The initial phase of this algorithm involves identification of distinguishable features, similar to the terrain marking. However, the vectors are subsequently drawn only for those objects, which are moving with respect to the terrain. The number of marked vectors for a moving object corresponds to the quality of extracted Harris features and the speed of moving object and therefore, defines the level of confidence and overall accuracy of the algorithm. We incorporated different adder models into a MATLAB-based Moving Object Tracking algorithm and processed their results to obtain movement vectors, as presented in Fig. 13. It can be seen that marker-pairs (red circles and yellow crosses) over static images exhibit zero displacement, which were subsequently filtered out to leave only the moving object vectors. Such moving object vectors can only be observed over the three moving vehicles, and are denoted as Obj-1, Obj-2 and Obj-3 (Fig. 13 (a)).

The results obtained through integration of different approximate adder modules into the parent algorithm were evaluated with the help of the number of markers placed on a moving object, and the vectors thus obtained. Comparative results of this analysis for different approximate adder models are presented as Fig. 14. The quality of these results can be observed to degrade significantly with the increase in error-probability of the corresponding adder.

V. CONCLUSION AND FUTURE WORK

Approximate Computing offers an alternative approach to effectively handle computationally-intensive error-tolerant applications. Derived through this motivation, we selected state-of-the-art approximate arithmetic designs and implemented them in Xilinx ISE 14.7 and Vivado with Virtex-7 FPGA based synthesis settings. Furthermore, we conducted mathematical error modeling of the designs to ascertain respective error probabilities. In addition, simulation-driven performance evaluations have also been conducted for a sample space of 12-bit 2-operand addition. Through analysis and comparison of these results with latest FPGA architecture in light of our proposed methodology, we have proposed a series of novel approximate adder designs focused on error-tolerant SWaP-constrained aerial applications. With our proposed designs, we have been able to achieve 9.9% power savings, when compared with state-of-the-art approximate adders, with high-accuracy performance suiting aerial reconnaissance applications, as verified through application-specific simulations. When deployed at system level, given the mathematical error analysis of our adder, adequate power savings can be achieved. In future, we intend to further optimize our design for video / image processing applications and subsequently analyze the suitability of approximate adders.
for time and safety-critical applications on-board resource-constrained platforms, like small sized mobile and remotely operated vehicles/robots. Another interesting extension to our research is the design of on-the-fly accuracy reconfigurable approximate adders for FPGA-based systems.

REFERENCES

[1] M. Pashaeifar, M. Kamal, A. Afzali-Kusha, and M. Pedram, “Approximate Reverse Carry Propagate Adder for Energy-Efficient DSP Applications,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 99, pp. 1–12, 2018.

[2] N. Zhu, W. Ling Goh, W. Zhang, K. Seng Yeo, and Z. Hui Kong, “Design of low-power high-speed truncation-error-tolerant adder and its application in digital signal processing,” IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 18, no. 8, pp. 1225–1229, Aug. 2010.

[3] V. Gupta, D. Mohapatra, A. R. Roy, and K. Roy, “Low-power digital signal processing using approximate adders,” IEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 32, no. 1, pp. 124–137, Jan. 2013.

[4] V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, and K. Roy, “IMPACT: IMPrecise adders for low-power approximate computing,” in Proc. IEEE/ACM Int. Symp. Low Power Electron. Design, Aug. 2011, pp. 409–414.

[5] A. B. Kahng and S. Kang, “Accuracy-configurable adder for approximate arithmetic designs,” in Proc. 49th Annu. Design Autom. Conf. DAC, 2012, pp. 820–825.

[6] B. S. Prabakaran, S. Rehman, M. A. Hanif, S. Ullah, G. Mazaheri, A. Kumar, and M. Shafique, “DeMAS: An efficient design methodology for building approximate adders for FPGA-based systems,” in Proc. Design, Autom. Test Eur. Conf. Exhib. (DATE), Mar. 2018, pp. 917–920.

[7] I. Kwon, R. Tessier, and J. Rose, “FPGA architecture: Survey and challenges,” Found. Trends Electron. Des. Autom., vol. 2, no. 2, pp. 135–253, 2008.

[8] T. Nomani and M. Mohsin, “A novel approximate adder design methodology with single LUT delay for fault-tolerant FPGA-based systems,” in Proc. 2nd Int. Conf. Latest Trends Electr. Eng. Comput. Technol. (INTELLECT), Nov. 2019, pp. 1–6.

[9] S. Ullah, S. Rehman, B. S. Prabakaran, F. Kriebel, M. A. Hanif, M. Shafique, and A. Kumar, “Area-optimized low-latency approximate multipliers for FPGA-based hardware accelerators,” in Proc. 55th ACM/ESDA/IEEE Design Autom. Conf. (DAC), Jun. 2018, p. 159.

[10] T. Ayhan, F. Kula, and M. Altun, “A power efficient system design methodology employing approximate arithmetic units,” in Proc. IEEE Comput. Soc. Annu. Symp. VLSI (ISVLSI), Jul. 2017, pp. 243–248.

[11] M. Shafique, W. Ahmad, R. Hafiz, and J. Henkel, “A low latency generic accuracy configurable adder,” in Proc. 52nd Annu. Design Autom. Conf. DAC, 2015, pp. 1–6.

[12] S. Mazahir, O. Hasan, R. Hafiz, M. Shafique, and J. Henkel, “An area-efficient consolidated configurable error correction for approximate hardware accelerators,” in Proc. 53rd Annu. Design Autom. Conf. DAC, 2016, p. 96.

[13] Q. Xu, T. Mytkowicz, and N. S. Kim, “Approximate computing: A survey,” IEEJ Des. Test, vol. 33, no. 1, pp. 1–8, Feb. 2016.

[14] J. Kaur and L. Sood, “Comparison between various types of adder topologies,” JCST, vol. 6, no. 1, pp. 62–66, Feb. 2015.

[15] S. A. Khan, Digital Design of Signal Processing Systems: A Practical Approach, Hoboken, NJ, USA: Wiley, 2011.

[16] T. V. P. Goloutsos, M. Erenguvac, S. Patel, and G. Reimann, “The art of deception: Adaptive precision reduction for area efficient physics accelerator,” in Proc. 40th Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO), 2007, pp. 394–406.

[17] S. Sidiroglou-Douskos, S. Misailovic, H. Hoffmann, and M. Rinard, “Managing performance vs. Accuracy trade-offs with loop perforation,” in Proc. 19th ACM SIGSOFT Symp. 13th Eur. Conf. Found. Softw. Eng. SIGSOFT/SIGOPS, 2011, pp. 124–134.

[18] A. Rahimi, L. Benini, and R. K. Gupta, “Spatial memoization: Concurrent instruction reuse to correct timing errors in SIMD architectures,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 60, no. 12, pp. 847–851, Dec. 2013.
TUAHA NOMANI received the B.E. and M.S. degrees in avionics engineering from the College of Aeronautical Engineering, National University of Sciences and Technology (NUST), Pakistan. He was associated with the Civil Aviation Authority (CAA), Pakistan, Turkish Aerospace Industries (TAI), Ankara, Turkey, and Lockheed Martin Production Facility, USA, for various projects. He is currently a Senior Scientific Researcher with the AI Processor Design Laboratory, Centre of Artificial Intelligence and Computing (CENTAIC), Pakistan. His research interests include FPGA-based solutions for aerial platforms and AI-oriented processing architectures.

MUJAHID MOHSIN received the M.S. degree (Hons.) in information security and the Ph.D. degree in computer and communication security from the National University of Sciences and Technology (NUST), Pakistan, in 2010 and 2017, respectively. He was a Researcher with CERN and also with the Cyber Defense and Network Assurability (CyberDNA) Center, University of North Carolina at Charlotte (UNCC), USA. His professional carrier stretches over 18 years and constitutes a diverse range of experiences, including design, development, maintenance, audit, and evaluation of electronics and information security systems with several multinational companies. He is currently an Associate Professor and the Head of the Communications Group with the College of Aeronautical Engineering (CAE), NUST. He has organized several international conferences, seminars and workshops in the domain of information security and the IoT. His research interests include approximate computing, FPGA optimizations, automated security analytics, the Internet of Things (IoT) security, and formal methods. He received the Best Faculty Certificate, in 2018, the Rector’s NUST High Achiever Award, in 2015, and the Ph.D. Scholarship from the Higher Education Commission, Pakistan, from 2013 to 2017.

ZAHID PERVAIZ received the B.E. degree in avionics engineering from the College of Aeronautical Engineering, National University of Sciences and Technology (NUST), Pakistan, and the M.S. and Ph.D. degrees in electrical and computer engineering from Purdue University, West Lafayette, IN, USA. He was awarded gold medal for securing first position in BE Avionics by NUST. During his graduate studies at Purdue University, he worked on NIST and NSF funded projects related to access control. He is currently an Associate Professor with the Department of Avionics Engineering, College of Aeronautical Engineering, NUST. His research interests include data privacy, distributed system security, and access control.

MUHAMMAD SHAFIQUE (Senior Member, IEEE) received the Ph.D. degree in computer science from the Karlsruhe Institute of Technology (KIT), Germany, in January 2011. He was with Streaming Networks Private Ltd., where he was involved in research and development of video coding systems for several years. He has been a Professor of computer architecture and robust energy-efficient technologies (CARE-Tech.) with the Institute of Computer Engineering, TU Wien, Austria, since November 2016. He holds one U.S. patent. He has authored or coauthored six books, more than ten book chapters, and over 200 papers in premier journals and conferences. He has given several keynote, invited talks, and tutorials. He has also organized many special sessions at premier venues. His research interests include computer architecture, power-energy efficient systems, robust computing, hardware security, brain-inspired computing, embedded artificial intelligence, hardware and system-level design for machine learning, emerging technologies and nanosystems, FPGAs, MPSoCs, and embedded systems. His research has a special focus on cross-layer modeling, design, optimization of computing and memory systems, and their deployment in use cases from the Internet-of-Things (IoT), cyber-physical systems (CPS), and ICT for development (ICT4D) domains. He is a Senior Member of the ACM, SIGARCH, SIGDA, SIGBED, and HIPEAC. He received the 2015 ACM/SIGDA Outstanding New Faculty Award, six gold medals in his educational career, several best paper awards and nominations with prestigious conferences, such as CODES+ISSS, DATE, DAC, and ICCAD, the Best Master Thesis Award, the DAC’14 Designer Track Best Poster Award, the IEEE TRANSACTIONS ON COMPUTER Feature Paper of the Month Awards, and the Best Lecturer Award. He has served on the TPC of the numerous prestigious IEEE/ACM conferences. He served as the Guest Editor for the IEEE Design and Test Magazine and the IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING.