IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 17 May 2024, accepted 20 May 2024, date of publication 23 May 2024, date of current version 31 May 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3404623

APPLIED RESEARCH

FPGA-Based Real-Time Object Detection and Classification System Using YOLO for Edge Computing

RASHED AL AMIN, (Graduate Student Member, IEEE), MEHRAB HASAN, VEIT WIESE, AND ROMAN OBERMAISSER
Institute for Embedded Systems, University of Siegen, 57076 Siegen, Germany
Corresponding author: Rashed Al Amin (rashed.amin@uni-siegen.de)

This work was supported in part by the TElepresence and collaboration of students in a mixed physical/virtual laboratory for CYber-physical systems (TECY) Project, funded by the Freiraum 2022 Program of Stiftung Innovation in der Hochschullehre.

ABSTRACT The leap forward in research progress in real-time object detection and classification has been dramatically boosted by including Embedded Artificial Intelligence (EAI) and Deep Learning (DL). Real-time object detection and classification with deep learning require many resources and computational power, which makes it more difficult to use deep learning methods on edge devices. This paper proposed a new, highly efficient Field Programmable Gate Array (FPGA) based real-time object detection and classification system using You Only Look Once (YOLO) v3 Tiny for edge computing. However, the proposed system has been instantiated with Advanced Driving Assistance Systems (ADAS) for evaluation. Traffic light detection and classification are crucial in ADAS to ensure drivers' safety. The proposed system used a camera connected to the Kria KV260 FPGA development board to detect and classify the traffic light. Bosch Small Traffic Light Dataset (BSTLD) has been used to train the YOLO model, and Xilinx Vitis AI has been used to quantify and compile the YOLO model. The proposed system can detect and classify traffic light signals from a high-definition (HD) video streaming in 15 frames per second (FPS) with 99% accuracy. In addition, it consumes only 3.5W power, demonstrating the ability to work on edge devices. The on-road experimental results represent fast, precise, and reliable detection and classification of traffic lights in the proposed system. Overall, this paper demonstrates a low-cost and highly efficient FPGA-based system for real-time object detection and classification.

INDEX TERMS FPGAs, object detection and classification, YOLO, edge computing.

I. INTRODUCTION

Deep Learning has been widely used for object detection and classification tasks [1] and the most common application of object detection and classification is in the domain of Advanced Driving Assistance Systems. The continual advancements in Computer Vision (CV) with integrated Artificial Intelligence (AI) decision-making and control have brought intelligent driving to the forefront of discussions in the realm of ADAS [2], [3], [4]. These systems are designed to assist drivers in their decision-making processes, offering coordination and notifications during unforeseen events while fulfilling the fundamental requirements of autonomous driving. Given the absence of vehicle-to-infrastructure communication in contemporary transportation systems, the detection and categorization of traffic lights have gained significant importance within the ADAS framework [5], [6].

Various techniques have been developed to enhance detection algorithms based on deep learning technologies, aiming to create a robust and comprehensive object detector [7]. Convolutional Neural Networks (CNN), rooted in deep learning, have made significant strides in object recognition and detection [8]. Ouyang et al. [5] employed CNN for traffic light detection using NVidia Jetson TX1/TX2, and Zhang et al. [9] implemented CNN for traffic light classification with FPGA. On the other hand, two-stage detectors

The associate editor coordinating the review of this manuscript and approving it for publication was Abdallah Kassem.

offer superior performance in both localization and recognition accuracy [9]. However, a significant drawback is their demand for substantial amounts of annotated data for training. Another algorithm, the Single Shot MultiBox Detector (SSD) [10], exhibits relatively poor performance in handling small objects [7]. The most noteworthy algorithm in this category is the You Only Look Once (YOLO) [11], which has several versions. Compact iterations of the YOLO algorithm have been developed to ensure efficient execution on hardware-constrained devices [8]. Wu et al. [12] and Abraham et al. [13] have presented traffic light detection systems using YOLO. In addition, Fernando et al. [14] and Huy et al. [15] have also demonstrated traffic light detection and classification using YOLO. Furthermore, various methods based on Lightweight [16], PID Controller [17], Salience-Sensitive Loss [18], vehicular ad-hoc networks [19], support vector machine [20], and Adaptive Background Suppression Filter [21] have been reported for traffic light detection and classification.

Edge computing and object detection are two critical components of modern technology, working hand in hand to provide efficient and real-time solutions in various applications using DL frameworks. However, DL models require high computational power and network bandwidth. Accelerating the DL model (e.g., YOLO) in edge devices can improve the system performance. Therefore, using field-programmable gate arrays (FPGAs) in object detection and classification systems enhances operational security and real-time computing ability while reducing prices.

FPGAs have proven to maintain safety critical systems with their remarkable computational ability. FPGAs are experiencing rapid growth in the domain of Artificial Intelligence (AI) acceleration, driven by their capacity for parallel processing and architectural optimizations [22]. FPGA-based implementations of deep learning models yield higher speed and accuracy with lower power consumption than software and image-based systems. To address the limitations associated with the low flexibility and accuracy of software and image-based algorithms, as well as the high power consumption of such methods, this paper presents a real-time and lightweight system for the detection and classification of traffic lights in autonomous vehicles. The YOLO v3 Tiny algorithm proves its suitability, mainly when deployed on FPGA boards. However, the newly introduced Xilinx Kria KV260 FPGA development board demonstrates its outstanding computational ability to accelerate deep learning algorithms [23], [24], [25], [26], [27], [28]. The synergistic pairing of the Xilinx Kria KV260 development board with the YOLO v3 algorithm is a compelling choice for real-time traffic light detection and classification on resource-constrained devices. In tandem with the detection and classification of traffic signals, the system also includes a critical module for speed control in compliance with the classification results. The main contributions of this paper are summarized as follows:,

1. This study presents a new system on the Xilinx Kria KV260 FPGA board, enhancing real-time object detection and classification with an optimized YOLO v3 Tiny deep learning model.
2. Instantiate the proposed object detection system to effectively identify and classify traffic light signals with precision and accuracy.
3. Evaluate the performance of the proposed system by comparing with state-of-the-art object detection system.

II. MOTIVATION AND RELATED WORK

Implementing YOLO-based object detection on FPGA brings forth both practical and theoretical implications. From a practical standpoint, it enables real-time processing of object detection tasks with minimal latency, rendering it suitable for applications necessitating swift decision-making, such as autonomous vehicles and surveillance systems. The FPGA's customizable hardware architecture facilitates optimized performance and resource efficiency, resulting in reduced power consumption and cost-effective solutions, particularly beneficial for embedded systems and IoT devices. On a theoretical level, this implementation underscores the potential synergy between deep learning algorithms like YOLO and hardware acceleration techniques such as FPGA, thereby laying the groundwork for further advancements in edge computing, where computational resources are constrained yet real-time processing is imperative. Furthermore, delving into the theoretical realm of optimizing deep learning models for FPGA architectures can drive progress in hardware-software co-design methodologies, ultimately fostering the development of efficient and scalable AI systems across diverse applications.

Numerous methods have endeavored to develop real-time object detection systems tailored for edge devices. Nevertheless, this section delves into an exploration of contemporary state-of-the-art object detection methodologies specifically designed for deployment on edge devices. Table 1 encapsulates a comprehensive summary of these object detection systems, providing a comparative overview of their respective features and performance metrics.

Several studies have proposed real-time object detection systems for edge devices utilizing Graphics Processing Unit (GPU)-based architectures. However, the significant power consumption associated with GPU-based systems presents a considerable bottleneck for their practical deployment [6], [12], [13], [16]. Notably, Abraham et al. [13] achieved the lowest power consumption among GPU-based traffic light detection systems. Their study focused on a traffic light detection system utilizing YOLO architecture and implemented on a Nvidia Tesla T4 GPU. The research introduces a modified YOLO model tailored for detecting traffic lights and signs. This model, based on a modified cross-stage partial YOLO v4 architecture, processes images captured by a camera sensor, leveraging a dataset comprising 1360 training data and

TABLE 1. State-of-the-art Object detection system.

Author/Criteria	Hardware	NN Model	Image Size	Target	Throughput (FPS)	Power Consumption (W)
Zhou et al. [20]	FPGA	HOG, SVM	1024×768	Traffic Light	60	-
Zhang et al. [30]	FPGA	CNN	320×176	Object	9	1.97
Wang et al. [29]	FPGA	CNN	32×32	Object	16	0.79
Abraham et al. [13]	GPU	YOLO	416×416	Traffic Light	29	70
Huu et al. [6]	GPU	YOLO	416×416	Traffic Light	30	250
Wu et al. [12]	GPU	YOLO	640×480	Traffic Light	54	450
Rzaev et al. [35]	FPGA	YOLO	224×224	Object	10	-
Wang et al. 36]	FPGA	YOLO	416×416	Object	26	-
Liu et al. [37]	FPGA	YOLO	640 × 640	Object	-	15.38
Pestana et al. [38]	FPGA	YOLO	768 × 576	Object	32	3.87
Zhang et al. [39]	FPGA	YOLO	1280×384	Object	36	-
Esen et al. [40]	FPGA	YOLO	1280×720	Object	16	-
Xin et al. [41]	FPGA	YOLO	416×416	Object	27	9.6
Heller et al. [31]	FPGA	YOLO	HD	Object	15	8
Corcoran et al. [32]	FPGA	YOLO	416 × 416	Object	69	15.4
Nguyen et al. [33]	FPGA	YOLO	HD	Object	125	26.4
Valadanzoj et al. [34]	FPGA	YOLO	416 × 416	Object	55	13.6

340 testing data, encompassing six types of traffic lights and 39 types of traffic signs. The network, implemented using the Darknet framework, attains a mean average precision of 79.77% at a processing speed of 29 FPS, accompanied by a power consumption of 70W.

Alternatively, CNN based object detection serves as another viable approach, characterized by notably low power consumption and high throughput [9], [29], [30]. However, a key bottleneck of CNN-based real-time object detection systems lies in their requirement for low input image sizes, necessitating substantial computational power for processing larger images. Addressing this challenge, Wang et al. [29] proposed a highly efficient CNN-based object detection system. Their work introduces an adaptive CNN edge computing platform tailored for target detection tasks, leveraging FPGA technology. This research capitalizes on the inherent pipeline architecture of FPGAs to expedite network computations, utilizing off-chip memory for storing network models and thereby obviating the need for resource-intensive tiling techniques. Moreover, an innovative online reconfigurable design is introduced, enabling real-time adjustments to network structure and parameters to accommodate diverse target recognition objectives. Implemented on a Spartan-6 FPGA platform, the system undergoes evaluation through pedestrian and vehicle classification tasks, achieving a detection speed of 16 frames per second (FPS) and a power consumption rate of 0.79 W, while attaining a classification accuracy of 96%. However, it is noteworthy that the system's evaluation was conducted with very low image sizes, resulting in comparatively lower accuracy.

Although YOLO-based object detection on FPGA offers a multitude of advantages including real-time processing, tailored optimization for FPGA architecture, low power consumption, high throughput, low latency, flexibility for various YOLO models, and resource efficiency, numerous YOLO-based object detection systems have emerged in recent years. However, to discern the state-of-the-art among these systems, several notable implementations warrant attention. Heller et al. [31] introduced an object detection system utilizing deep learning on embedded edge devices, focusing on maritime object detection with the Xilinx Kria KV260 Vision AI Kit. Their study involved training and evaluating multiple YOLO neural networks of varying sizes and architectural specifications, incorporating structured pruning techniques such as sparsifying to reduce network size while preserving detection performance. The proposed deployments showcased promising outcomes, achieving an inference speed of 90 FPS with only a marginal 2.4% degradation in mean average precision for high-definition input images. Nonetheless, while exhibiting enhanced throughput and efficiency, the accuracy of this method necessitates further refinement.

Corcoran et al. [32] proposed a streaming architecture toolflow to accelerate YOLO models on FPGA devices, employing a deeply pipelined on-chip design for YOLO accelerators. These accelerators, generated using an

automated toolflow, incorporate novel hardware components supporting YOLO operations in a dataflow manner, along with off-chip memory buffering to mitigate on-chip memory limitations. Their approach achieves a throughput of 69 FPS for input images of 416 × 416, consuming 15.4W power. However, power consumption of this study remains relatively high, and accuracy metrics are not provided, posing potential limitations.

Nguyen et al. [33] present an FPGA-based design for YOLOv4 network tailored for flying-object detection, addressing challenges of limited floating-point resources while aiming to maintain accuracy, real-time performance, and energy efficiency. They curated a suitable dataset of flying objects for network implementation, training, and fine-tuning, adapting YOLO networks for FPGA deployment. Evaluating on the ZCU104 FPGA kit, they achieved 125FPS for HD input images, consuming 26.4W power. However, high throughput is accompanied by elevated power consumption and potentially compromised accuracy, suggesting potential bottlenecks.

Valadanzoj et al. [34] introduced a high-speed YOLO hardware accelerator tailored for self-driving automotive applications on FPGA. Their approach involved utilizing 8-bit and 5-bit fixed-point formats for data and weights to conserve resources and memory. To address accuracy concerns, a Genetic Algorithm was employed to optimize the decimal point positions across different network layers. Furthermore, a technique enabling simultaneous multiplications with distinct operands using a single DSP block was presented, enhancing network execution speed. Evaluation on the Xilinx Zynq ZC706 FPGA platform yielded a throughput of 55 FPS with 79% accuracy for a given input image, consuming 13.6W of power. Nonetheless, similar to previous studies, the high throughput was accompanied by elevated power consumption and compromised accuracy, potentially posing a limitation.

Besides of these large NN based architecture, several alternative methods have been reported for real-time object detection, including lightweight approaches [16] and combinations of Blob, Histogram of Oriented Gradients (HOG), and Support Vector Machine (SVM) [20]. However, the performance of these methods still requires improvement compared to other state-of-the-art object detection systems.

The majority of prior research has predominantly focused on throughput and accuracy metrics, often neglecting the critical aspects of power consumption and cost, which are pivotal considerations for edge computing systems. Moreover, some studies have utilized highly expensive FPGA boards, limiting their practical applicability, particularly for large-scale systems or ADAS. Unfortunately, current state-of-the-art object detection systems still require improvement across all performance metrics, including throughput, accuracy, power consumption, and cost-effectiveness. To address these constraints, this paper introduces a novel object detection system for FPGA utilizing YOLO. Leveraging a $250 FPGA board (Xilinx Kria KV-260) with minimal power consumption, the proposed system aims to maintain accuracy and throughput while operating within cost constraints. However, achieving a trade-off among these performance evaluation metrics with a low-cost FPGA board presents significant challenges. The Kria KV-260, being a newly launched FPGA board with limited memory and computational power, may pose a bottleneck for deploying large neural network models. Thus, the proposed system optimizes the YOLO model to fit within the FPGA board's constraints. Additionally, evaluating the proposed system through on-road tests and using street photos and videos is a crucial aspect of this study, providing real-world validation of its effectiveness.

III. IMPLEMENTATION

The initiation of the study's implementation process commences with the establishment of the host machine environment and the FPGA development board. This implementation procedure encompasses multiple sequential stages, encompassing environment configuration, dataset preparation, model training, model conversion, and deployment of the trained model into the FPGA board, all aimed at creating a robust and dependable object detection and classification system. The overall architecture of the proposed system is illustrated in Figure 1.

A. DATASET

The dataset has been prepared according to the YOLO format to train a YOLO model in the darknet framework. The images from the collected dataset were annotated with the bounding boxes surrounding the traffic lights that must be detected. The annotations consist of the traffic light class and coordinates of the center on the X-axis, coordinates of the center on the Y-axis, height, and width. A class name was allocated to every annotation according to the traffic light class (Red, Green, Yellow, Left Green, Right Green, Left Red, Right Red, and No Light). The pixel values of bounding box coordinates (x, y, width, height) were transformed to normalized values between 0 and 1 relative to the dimensions of the traffic light image to improve the model's convergence and robustness. However, to train the YOLO model, a corresponding text file has been generated for each traffic light image in the BSTLD dataset. The information about the traffic light in the text file maintains a predefined sequence so that during the training, the darknet framework can extract the information correctly and train the YOLO model with accurate data.

The Bosch Small Traffic Light Dataset (BSTLD) [42] contains a total of 13,427 camera photos, each with a size of 1280 × 720 pixels. Additionally, the collection includes around 24,000 annotated traffic light signals. The images in the training set depict a variety of obstacles that may be encountered when driving in urban environments. However, 5094 images from the BSTLD images were used for training purposes in this study, and among those images, 1019 images were split to make a test set.

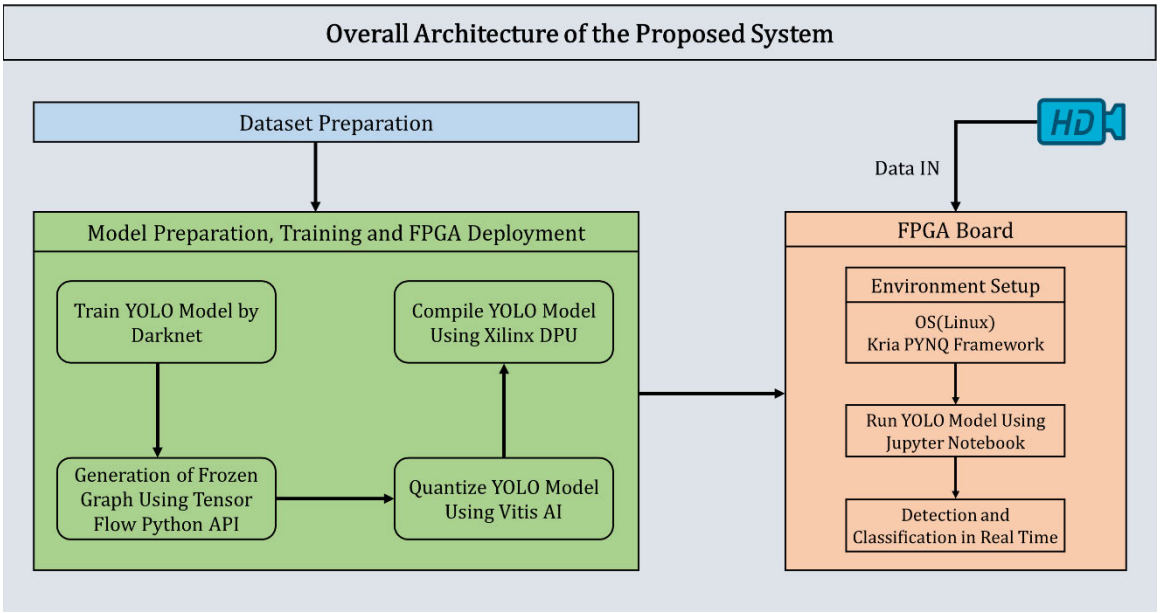

FIGURE 1. Overall architecture of the proposed system.

B. MODEL PREPARATION AND FPGA DEPLOYMENT

1) MODEL TRAINING THROUGH DARKNET

Darknet framework serves the dual purpose of training neural networks and subsequently processing images or video frames using deep neural networks. However, to train the YOLO model, this study considers the darknet framework due to its real-time object detection architecture. Darknet mainly employs 53 convolutional layers in 3×3 and 1×1 convolutional filters to extract features and reduce output [43]. Besides, It makes predictions by utilizing max pooling. YOLO framework comes with a configuration file that needs to be modified according to the project-specific requirements. These configurations play a crucial role in determining the detection capabilities of the model. YOLOv3 configuration files have been changed for traffic light detection and classification, including the number of classes, anchor boxes, and other hyperparameters. Initially, both the batch and subdivisions were set to one. To train efficiently, the batch was set to 64, and the subdivision was set to 8 by considering the volume of images in the BSTLD and the capacity of the host machine.

The proposed YOLOv3 Tiny model utilizes 3×3 convolution layers with a stride size of 1 to extract features through a feedforward structure. The initial convolution layer takes as input an image of size 416×416, and each convolution layer incorporates a padding size of 1. Furthermore, it employs Max pooling layers to down sample data within the convolution layers. Bounding box predictions are made at two distinct feature map scales and combined with an up sampled 13×13 feature map. YOLOv3 Tiny detects 3 boxes per grid cell with 5 bounding boxes. Since this study considers 8 traffic light classes, the number of filters within the convolutional layer has been set to 39 in the configuration file. The YOLO model was trained with weights from a pre-trained model from the darknet to maximize efficiency. This step is followed by fine-tuning the YOLOv3 Tiny network with the dataset by running the dataset through the neural network and modifying the parameters to minimize the loss. Table 2 represents the optimized YOLO v3 Tiny model architecture for the proposed system.

2) GENERATION OF FROZEN GRAPH

Freezing the graph is a process that combines the architecture of the model and weights into a single file called a frozen graph. The darknet binary weights were converted into the TensorFlow variables. The yolov3 Tiny architecture has been built in TensorFlow with the proper configuration file and the appropriate weight file generated from the Darknet framework.

The architectural framework of the model has been replicated with input and output nodes along with other associated parameters. TensorFlow has built the YOLO model and generated the frozen graph with the neural network layers, the determination of activation functions, and the configuration of input-output functions in this process by using the configuration and weights files. In addition, the frozen graph has been optimized by the TensorFlow API. The procedure for generating a frozen graph illustrated in Fig. 2.

3) QUANTIZATION

Vitis-AI quantizer provides a function to convert the 32-bit floating-point weights and activations to decrease computational complexity and maintain prediction accuracy. The frozen graph has been investigated by the Vitis-AI to find out the number of input and output nodes of the trained YOLO

TABLE 2. Modified YOLO architecture of the proposed system.

Layer	Type	Activation Function	No. of Filters	Size/Stride	Output
1	Convolutional	ReLU	16	3 × 3/1	416 × 416 × 16
2	Maxpool			2 × 2/2	208 × 208 × 16
3	Convolutional	ReLU	32	3 × 3/1	208 × 208 × 32
4	Maxpool			2 × 2/2	104 × 104 × 32
5	Convolutional	ReLU	64	3 × 3/1	104 × 104 × 64
6	Maxpool			2 × 2/2	52 × 52 × 64
7	Convolutional	ReLU	128	3 × 3/1	52 × 52 × 128
8	Maxpool			2 × 2/2	26 × 26 × 128
9	Convolutional	ReLU	256	3 × 3/1	26 × 26 × 256
10	Maxpool			2 × 2/2	13 × 13 × 256
11	Convolutional	ReLU	512	3 × 3/1	13 × 13 × 512
12	Maxpool			2 × 2/2	13 × 13 × 512
13	Convolutional	ReLU	1024	3 × 3/1	13 × 13 × 1024
14	Convolutional	ReLU	256	1 × 1/1	13 × 13 × 256
15	Convolutional	ReLU	512	3 × 3/1	13 × 13 × 512
16	Convolutional	Linear	39	1 × 1/1	13 × 13 × 39
17	YOLO				
18	Route				
19	Convolutional	ReLU	128	1 × 1/1	13 × 13 × 128
20	Up-sampling			2	26 × 26 × 128
21	Route				
22	Convolutional	ReLU	256	3 × 3/1	13 × 13 × 256
23	Convolutional	Linear	39	1 × 1/1	13 × 13 × 39
24	YOLO				

model and their names. To calibrate the trained YOLO model before quantization, 250 random images have been selected from the test set. However, the model has been quantized into INT8 fixed point formats that require less memory bandwidth, improving power efficiency and computing speed.

FIGURE 2. Frozen graph generation process.

4) MODEL COMPILATION

The Vitis-AI compiler has been used to compile the quantized model into the Kria KV260 FPGA board using Xilinx DPUCZDX8G. The compilation process has been performed using the Xilinx Intermediate Representation (XIR) based compiler of Vitis-AI. Initially, the quantized model was used as the input for the compiler, and then the compiler transformed the model into the XIR graph. The graph was broken into different subgraphs by the compiler, and optimization was applied to the subgraphs. Besides the compiler generated the instruction stream and attached it to the DPU subgraph. However, the instructions and required information for Vitis-AI Runtime (VART) has serialized to compile the model into '.xmodel' format. Fig. 3 illustrates the interconnection between the DPU and the processor of the Kria KV260, with a 32-bit memory-mapped AXI interface employed to establish the links between the processor and the DPU. In the compilation process, the proper fingerprint of the DPU architecture has been specified to make the compiled model compatible with the DPU. The compiled model has been deployed to the Kria KV260 FPGA board to detect and classify the traffic lights.

IV. EXPERIMENTAL EVALUATION AND DISCUSSION

The evaluation of the proposed system commenced with a comprehensive software analysis. This preliminary phase involved rigorous testing and validation of the system's algorithms and functionalities in TensorFlow. Once the software results met the predefined criteria and demonstrated satisfactory performance, the proposed system has been transitioned to hardware deployment for further evaluation. The hardware assessment phase involved implementing the system on the target FPGA platform and subjecting it to real-world testing scenarios. This stage aimed to validate the system's performance under actual hardware constraints and operational conditions, ensuring its effectiveness and reliability in practical applications. By adopting a systematic approach that encompasses both software analysis and hardware evaluation, the proposed system can undergo thorough validation across various dimensions, thereby instilling confidence in its capabilities and suitability for deployment in real-world scenarios.

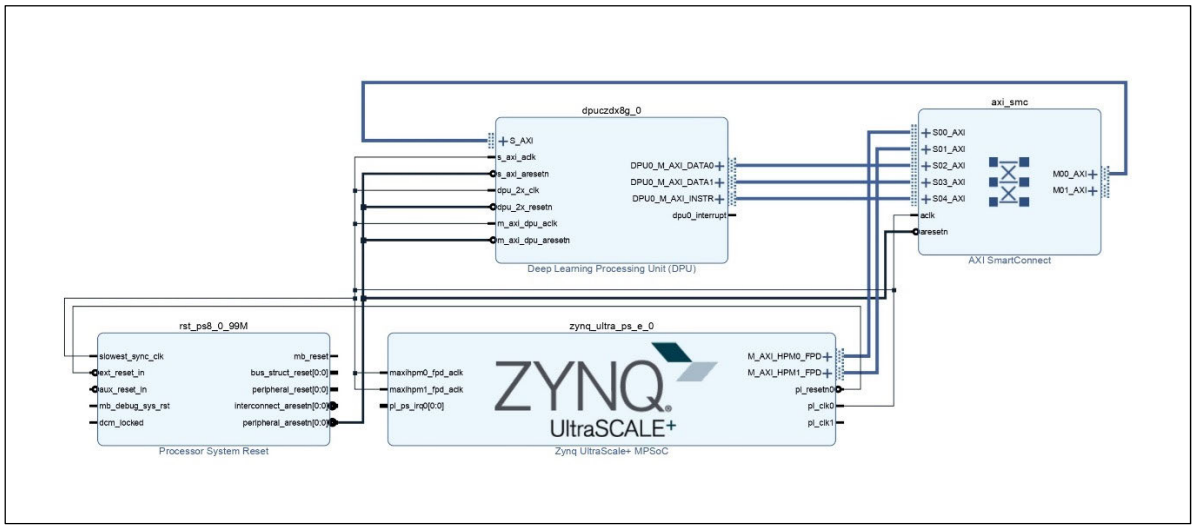

FIGURE 3. Interconnection between Xilinx DPU and the Kria KV260 processor.

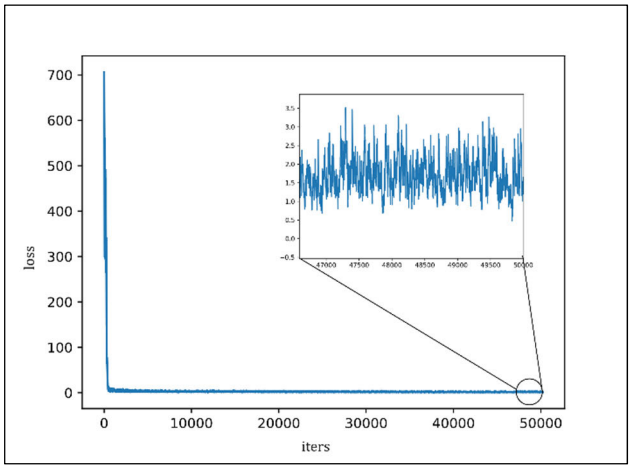

FIGURE 4. Loss vs. Iteration graph of the trained YOLO model.

A. EXPERIMENTAL SETUP

The YOLO model preparation, training, and FPGA deployment are carried out by a Linux host machine with Intel(R) Core (TM) i7-9750H, 2.66GHZ, and 16GB of RAM. The simulation environment is implemented using the Darknet framework and Xilinx Vitis-AI. One environment is for training the model, and another is for the model conversion according to the requirements of the development board. For the hardware evaluation phase, the Xilinx KV-260 FPGA board was utilized. This board features the Xilinx Zynq UltraScale+ MPSoC, which integrates a quad-core ARM Cortex-A53 processor alongside a programmable FPGA fabric. Additionally, the board offers a diverse array of interfaces and peripherals tailored for camera and display connectivity, facilitating seamless integration into object detection systems. Leveraging the computational capabilities of the ARM processor and the flexibility of the FPGA fabric, the Xilinx KV-260 FPGA board serves as an ideal hardware platform for implementing and evaluating the proposed object detection system. Its versatile architecture and robust feature set enable comprehensive hardware assessment, ensuring the system's performance and functionality align with the intended objectives and requirements.

B. SOFTWARE EVALUATION

The training process extended to approximately 50,200 iterations. Initially, the loss exceeded 700%, signifying a high error rate. Subsequently, as the training progressed, the

loss exhibited a rapid decline. Until around 1,000 iterations, there was a notable fluctuation in the loss. However, after approximately 1,500 iterations, the loss experienced a significant reduction, and the fluctuation rate decreased substantially compared to the earlier stages of training. Notably, the generated weight file at the 50,000th iteration has been utilized for traffic light detection and classification, designating this segment as the region of interest in the graph. The loss and iteration graph, as presented in Fig. 4, showcases this specific region, demonstrating that the loss stabilizes at approximately 1%.

The accuracy versus iteration graph is represented in Fig. 4. At the outset of the training process, the accuracy was notably low. However, with an increase in the number of iterations, accuracy improved considerably, particularly up to around 1,000 iterations. After approximately 1,500 iterations, the accuracy exhibited minimal fluctuation. Fig. 5 highlights the region of interest with the model's accuracy. However, it stabilizes at an very high level of accuracy of approximately 99%.

This study has investigated the efficacy of training and optimizing a YOLO v3 Tiny model for real-time traffic light detection and classification. With an outstanding 99% overall accuracy achieved during model training, the model showcases remarkable proficiency in accurately identifying

TABLE 3. Comparison of the proposed system with different FPGA based object detection system.

Author/Criteria	Wang et al. [29]	Heller et al. [31]	Corcoran et al. [32]	Nguyen et al. [33]	Valadanzoj et al. [34]	This Work
Year	2023	2022	2023	2024	2024	2024
NN Model	CNN	YOLO V4 Tiny	YOLO V3 Tiny	YOLO V4 Tiny	YOLO V4 Tiny	YOLO V3 Tiny
FPGA	Spartan-6	Kria KV 260	VCU110	ZCU104	ZC706	Kria KV260
Test Image Size	32×32	HD	416 × 416	HD	416 × 416	HD
Accuracy (%)	96	75	-	78	79	99
Throughput (FPS)	16	15	69	125	55	15
Power (W)	0.79	8	15.4	26.4	13.6	3.5
Efficiency (FPS/W)	20.25	1.87	4.5	4.7	4	4.2

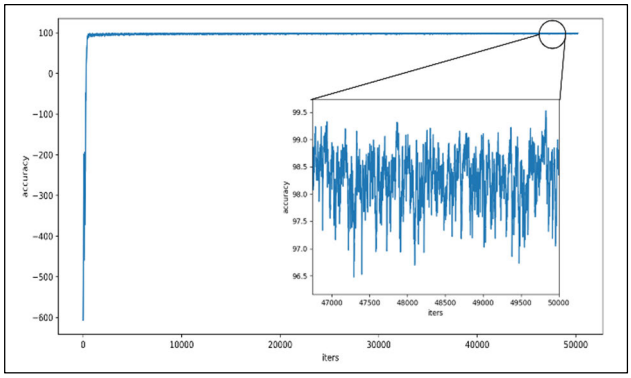

FIGURE 5. Accuracy vs. Iteration graph of the trained YOLO model.

objects amidst complex backgrounds. An essential step in this process was the successful conversion of darknet weights into TensorFlow format, enabling the application of quantization and compilation techniques. This conversion not only optimized system memory utilization but also preserved accuracy. Notably, the model's loss remained at approximately 1% within the region of interest, particularly at 50,000 iterations, underscoring its suitability for efficient operational deployment.

C. HARDWARE EVALUATION

The YOLO model underwent rigorous testing and evaluation on the Kria KV260 FPGA board. Impressively, the proposed system yielded highly satisfactory results. During an on-road experiment conducted on the streets of Siegen, Germany, the system demonstrated real-time capabilities by successfully detecting and classifying traffic lights from a high-definition video stream. In addition to evaluating the system using the BSTLD test images, this study also subjected the system to testing using 30 on-road experimental images with a resolution of 720 × 1280. Furthermore, five videos with HD resolution has been utilized, each with a duration of 2 minutes and a frame rate of 60 fps, resulting in a total of 7200 frames captured from street environments. These real-world on-road experiment images and videos provide valuable insights into the system's performance under varying environmental conditions and dynamic scenarios, further validating its effectiveness and reliability in practical deployment scenarios. During the experiment, the system exhibited an impressive processing time of 1.996 seconds to detect and classify 30 images, achieving a commendable speed of 15 FPS. Fig. 6 visually portrays the outcomes of traffic light detection and classification by the proposed system. To further assess the accuracy of annotation, segmentation, and object detection, the average Intersection over Union (IOU) has been calculated for various regions. Notably, the proposed model attained an average IOU of 36%, signifying its robust performance in these critical aspects.

D. CMPARISON WITH OTHER RELATED WORKS

In order to evaluate the performance discrepancy between the architecture delineated in this study and other heterogeneous architectures, a comparative analysis was undertaken by comparing the outcomes of the proposed system with those of analogous works documented in recent literature. This assessment aimed at appraising the efficacy of the system architecture elucidated in this research endeavor. Table 3 encapsulates the outcomes of various FPGA-based object detection systems alongside the proposed system. The findings distinctly illustrate that the CNN-based approach exhibits heightened efficiency owing to the diminutive size of the test images. Corcoran et al. [32] and Valadanzoj et al. [34] denote elevated throughput albeit accompanied by increased power consumption and accuracy. Conversely, Heller et al. [31] and Valadanzoj et al. [34] showcase a harmonized trade-off among throughput, power consumption, and efficiency. For an equitable comparison, the proposed system is compared with the work of Heller et al. [31], given the congruent utilization of FPGA boards and test image sizes in the experimentation. Fig. 7 delineates the performance evaluation of our system with the top notched state-of-the-art systems. Remarkably, the performance of the proposed system evinces a better performance across all evaluation metrics, manifesting a 24% enhancement in accuracy, 56% reduction in power consumption, and 55% augmenta-

FIGURE 6. Detection and classification results of the proposed system.

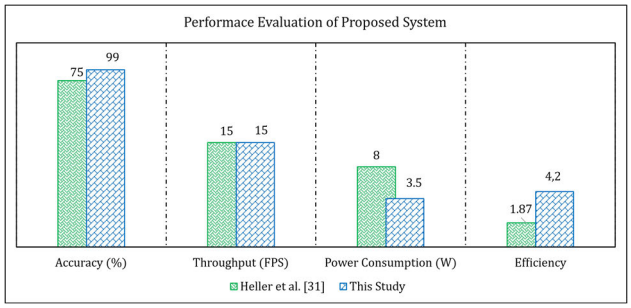

FIGURE 7. Performance evaluation of the proposed system.

tion in efficiency. Overall, the proposed system demonstrates satisfactory performance compared to other systems. The robustness and reliability of the YOLOv3 Tiny model in real-time scenarios are underscored by its deployment on the Kria KV260 and efficient utilization of the Xilinx DPU. However, isolated misdetections were encountered during the testing phase, albeit their incidence was minimal relative to the total number of objects tested.

V. CONCLUSION

An FPGA-based general YOLO model, tailored for object detection and classification in edge computing devices, has been introduced. This model has been carefully optimized to meet the stringent requirements of edge systems, encompassing limitations in hardware resources, the need for high accuracy, and real-time processing speed. In light of the system's inherent resource constraints and its overall performance, it is justifiable to conclude that both the model and the entire system have demonstrated their efficiency, reliability, and speed, particularly within the domain of Advanced Driver Assistance Systems (ADAS) and edge computing.

One noteworthy feature of the developed system is its versatility. It possesses the capability to detect not only traffic lights but also a wide range of other object types. This flexibility arises from the system's adaptability through the training of the YOLO model with suitable datasets. Consequently, ample room remains for further exploration and enhancement of the YOLO model. Architectural modifica-

tions and optimization efforts can be undertaken to achieve improved performance.

Additionally, it is worth considering the integration of a fault tolerance mechanism within the system to handle unexpected issues and ensure robustness. A valuable avenue for future work is the assessment of system reliability through long-term operation, observing its behavior and patterns across various real-world scenarios. Such investigations will contribute to the continuous development and refinement of the system's performance and reliability.

REFERENCES

[1] X. Xu, X. Zhang, B. Yu, X. S. Hu, C. Rowen, J. Hu, and Y. Shi, "DAC-SDC low power object detection challenge for UAV applications," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 2, pp. 392–403, Feb. 2021.

[2] H. Xu, M. Guo, N. Nedjah, J. Zhang, and P. Li, "Vehicle and pedestrian detection algorithm based on lightweight YOLOv3-promote and semi-precision acceleration," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 19760–19771, Oct. 2022.

[3] L. Fang, Q. Jiang, J. Shi, and B. Zhou, "TPNet: Trajectory proposal network for motion prediction," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6796–6805.

[4] X. Chang, H. Pan, W. Sun, and H. Gao, "YolTrack: Multitask learning based real-time multiobject tracking and segmentation for autonomous vehicles," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 12, pp. 5323–5333, Dec. 2021.

[5] Z. Ouyang, J. Niu, Y. Liu, and M. Guizani, "Deep CNN-based real-time traffic light detector for self-driving vehicles," *IEEE Trans. Mobile Comput.*, vol. 19, no. 2, pp. 300–313, Feb. 2020.

[6] T. H. P. Tran and J. W. Jeon, "Accurate real-time traffic light detection using YOLOv4," in *Proc. IEEE Int. Conf. Consum. Electron. Asia (ICCE-Asia)*, Nov. 2020, pp. 1–4.

[7] H. Naimi, "Traffic sign and light detection using deep learning for automotive applications," Ph.D. thesis, Dept. Elect. Comput. Eng., Univ. Windsor, Windsor, ON, Canada, 2021.

[8] P. Adarsh, P. Rathi, and M. Kumar, "YOLOv3-tiny: Object detection and recognition using one stage improved model," in *Proc. 6th Int. Conf. Adv. Comput. Commun. Syst. (ICACCS)*, Mar. 2020, pp. 687–694.

[9] J. Zhang, F. Zhang, M. Xie, X. Liu, and T. Feng, "Design and implementation of CNN traffic lights classification based on FPGA," in *Proc. IEEE 4th Int. Conf. Electron. Inf. Commun. Technol. (ICEICT)*, Aug. 2021, pp. 445–449.

[10] W. Liu, "SSD: Single shot multibox detector," in *Proc. 14th Eur. Conf.* Amsterdam, The Netherlands: Springer, Oct. 2016, pp. 21–37.

[11] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.

[12] S. Wu, N. Amenta, J. Zhou, S. Papais, and J. Kelly, "AUToLights: A robust multi-camera traffic light detection and tracking system," 2023, *arXiv:2305.08673*.

[13] A. Abraham, D. Purwanto, and H. Kusuma, "Traffic lights and traffic signs detection system using modified you only look once," in *Proc. Int. Seminar Intell. Technol. Appl. (ISITIA)*, Surabaya, Indonesia, 2021, pp. 141–146.

[14] W. H. D. Fernando and S. Sotheeswaran, "Automatic road traffic signs detection and recognition using 'you only look once' version 4 (YOLOv4)," in *Proc. Int. Res. Conf. Smart Comput. Syst. Eng. (SCSE)*, vol. 4, Sep. 2021, pp. 38–43.

[15] H. K. Hua, K. H. Nguyen, L.-D. Quach, and H. N. Tran, "Traffic lights detection and recognition method using deep learning with improved YOLOv5 for autonomous vehicle in ROS2," in *Proc. 8th Int. Conf. Intell. Inf. Technol.*, Feb. 2023, pp. 117–122.

[16] Z. Yao, Q. Liu, Q. Xie, and Q. Li, "TL-detector: Lightweight based real-time traffic light detection model for intelligent vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 9, pp. 9736–9750, Sep. 2023.

[17] S. Shrivastava, A. Somthankar, V. Pandya, and M. Patil, "Implementation of a pid controller for autonomous vehicles with traffic light detection in CARLA," in *Intelligent Computing and Networking*. Cham, Switzerland: Springer, 2023.

[18] R. Greer, A. Gopalkrishnan, J. Landgren, L. Rakla, A. Gopalan, and M. Trivedi, "Robust traffic light detection using salience-sensitive loss: Computational framework and evaluations," 2023, *arXiv:2305.04516*.

[19] E. Al-Ezaly, H. M. El-Bakry, A. Abo-Elfetoh, and S. Elhishi, "An innovative traffic light recognition method using vehicular ad-hoc networks," *Sci. Rep.*, vol. 13, no. 1, p. 4009, Mar. 2023.

[20] Y. Zhou, Z. Chen, and X. Huang, "A system-on-chip FPGA design for real-time traffic signal recognition system," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2016, pp. 1778–1781.

[21] Z. Shi, Z. Zou, and C. Zhang, "Real-time traffic light detection with adaptive background suppression filter," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 3, pp. 690–700, Mar. 2016.

[22] K. Sruthi and R. Nandakumar, "AI/ML-based object detection on FPGA SoC," in *Proc. Int. Conf. Commun., Electron. Digit. Technol.* Cham, Switzerland: Springer, 2023, pp. 479–487.

[23] S. Kalapothas, G. Flamis, and P. Kitsos, "Efficient edge-AI application deployment for FPGAs," *Information*, vol. 13, no. 6, p. 279, May 2022.

[24] K. Shi, M. Wang, X. Tan, Q. Li, and T. Lei, "Efficient dynamic reconfigurable CNN accelerator for edge intelligence computing on FPGA," *Information*, vol. 14, no. 3, p. 194, Mar. 2023.

[25] G. Rathod, P. Shah, R. Gajjar, M. I. Patel, and N. Gajjar, "Implementation of real-time object detection on FPGA," in *Proc. 7th Int. Conf. Trends Electron. Informat. (ICOEI)*, Apr. 2023, pp. 235–240.

[26] P. Dhilleswararao, S. Boppu, M. S. Manikandan, and L. R. Cenkeramaddi, "Efficient hardware architectures for accelerating deep neural networks: Survey," *IEEE Access*, vol. 10, pp. 131788–131828, 2022.

[27] S. C. Magalhães, F. N. dos Santos, P. Machado, A. P. Moreira, and J. Dias, "Benchmarking edge computing devices for grape bunches and trunks detection using accelerated object detection single shot multibox deep learning models," *Eng. Appl. Artif. Intell.*, vol. 117, Jan. 2023, Art. no. 105604.

[28] M. Baczmanski, M. Wasala, and T. Kryjak, "Implementation of a perception system for autonomous vehicles using a detection-segmentation network in SoC FPGA," in *Proc. Int. Symp. Appl. Reconfigurable Comput.* Cham, Switzerland: Springer, 2023, pp. 200–211.

[29] Y. Wang, Y. Liao, J. Yang, H. Wang, Y. Zhao, C. Zhang, B. Xiao, F. Xu, Y. Gao, M. Xu, and J. Zheng, "An FPGA-based online reconfigurable CNN edge computing device for object detection," *Microelectron. J.*, vol. 137, Jul. 2023, Art. no. 105805.

[30] Z. Zhang, M. A. P. Mahmud, and A. Z. Kouzani, "FitNN: A low-resource FPGA-based CNN accelerator for drones," *IEEE Internet Things J.*, vol. 9, no. 21, pp. 21357–21369, Nov. 2022.

[31] D. Heller, M. Rizk, R. Douguet, A. Baghdadi, and J.-P. Diguet, "Marine objects detection using deep learning on embedded edge devices," in *Proc. IEEE Int. Workshop Rapid Syst. Prototyping (RSP)*, Shanghai, China, Oct. 2022, pp. 1–7.

[32] A. Montgomerie-Corcoran, P. Toupas, Z. Yu, and C.-S. Bouganis, "SATAY: A streaming architecture toolflow for accelerating YOLO models on FPGA devices," in *Proc. Int. Conf. Field Program. Technol. (ICFPT)*, Dec. 2023, pp. 179–187.

[33] D.-D. Nguyen, D.-T. Nguyen, M.-T. Le, and Q.-C. Nguyen, "FPGA-SoC implementation of YOLOv4 for flying-object detection," *J. Real-Time Image Process.*, vol. 21, no. 3, p. 63, May 2024.

[34] Z. Valadanzoj, H. Daryanavard, and A. Harifi, "High-speed YOLOv4-tiny hardware accelerator for self-driving automotive," *J. Supercomput.*, vol. 80, no. 5, pp. 6699–6724, Mar. 2024.

[35] E. Rzaev, A. Khanaev, and A. Amerikanov, "Neural network for real-time object detection on FPGA," in *Proc. Int. Conf. Ind. Eng., Appl. Manuf. (ICIEAM)*, Sochi, Russia, May 2021, pp. 719–723.

[36] G. Wang, H. Ding, B. Li, R. Nie, and Y. Zhao, "Trident-YOLO: Improving the precision and speed of mobile device object detection," *IET Image Process.*, vol. 16, no. 1, pp. 145–157, Jan. 2022.

[37] M. Liu, S. Luo, K. Han, B. Yuan, R. F. DeMara, and Y. Bai, "An efficient real-time object detection framework on resource-constricted hardware devices via software and hardware co-design," in *Proc. IEEE 32nd Int. Conf. Application-Specific Syst., Archit. Processors (ASAP)*, Jul. 2021, pp. 77–84.

[38] D. Pestana, P. R. Miranda, J. D. Lopes, R. P. Duarte, M. P. Véstias, H. C. Neto, and J. T. De Sousa, "A full featured configurable accelerator for object detection with YOLO," *IEEE Access*, vol. 9, pp. 75864–75877, 2021.

[39] J. Zhang, L. Cheng, C. Li, Y. Li, G. He, N. Xu, and Y. Lian, "A low-latency FPGA implementation for real-time object detection," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2021, pp. 1–5.
[40] F. Esen, A. Degirmenci, and O. Karal, "Implementation of the object detection algorithm (YOLOV3) on FPGA," in *Proc. Innov. Intell. Syst. Appl. Conf. (ASYU)*, Elazig, Turkey, Oct. 2021, pp. 1–6.
[41] J. Xin, M. Cha, L. Shi, C. Long, H. Li, F. Wang, and P. Wang, "Lightweight convolutional neural network of YOLOv3-tiny algorithm on FPGA for target detection," in *Proc. IEEE Int. Conf. Comput. Sci., Artif. Intell. Electron. Eng. (CSAIEE)*, Aug. 2021, pp. 65–70.
[42] K. Behrendt, L. Novak, and R. Botros, "A deep learning approach to traffic lights: Detection, tracking, and classification," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 1370–1377.
[43] T. Diwan, G. Anirudh, and J. V. Tembhurne, "Object detection using YOLO: Challenges, architectural successors, datasets and applications," *Multimedia Tools Appl.*, vol. 82, no. 6, pp. 9243–9275, Mar. 2023.

RASHED AL AMIN (Graduate Student Member, IEEE) received the B.Sc. degree in electrical and electronics engineering from the University of Dhaka, Bangladesh, and the M.Sc. degree in mechatronics from the University of Siegen, Germany, where he is currently pursuing the Ph.D. degree with the Chair of Embedded Systems. He is also a Scientific Associate and a Lecturer with the Chair of Embedded Systems, University of Siegen. His research interests include hardware for artificial intelligence, reconfigurable computing, and FPGA architecture. His goal is to design the next generation FPGA architecture to optimize deep learning workloads. He received the Deans Award during the master's study.

MEHRAB HASAN received the B.Sc. degree from American International University, Bangladesh, and the M.Sc. degree in mechatronics from the University of Siegen, Germany, in 2023. His research interests include FPGA design, hardware accelerator, and multiprocessor architectures.

VEIT WIESE received the Dipl.-Ing. degree in electrical engineering from the University of Siegen, Germany, in 2015. Later on, he was a Project Manager with Qualigon GmbH, until 2017. He is a Project Manager and a Lecturer with the Institute of Embedded Systems, University of Siegen. His research interests include fault tolerance, safety-critical embedded systems, structural health monitoring, field-programmable gate arrays, and artificial intelligence.

ROMAN OBERMAISSER received the master's degree in computer sciences from Vienna University of Technology, in 2001, and the Ph.D. degree in computer science from Vienna University of Technology, in February 2004, with Prof. Hermann Kopetz, as a Research Advisor. In July 2009, he has received the Habilitation ("Venia docendi") Certificate for technical computer science. He is a Full Professor with the Institute for Embedded Systems, University of Siegen. He wrote a book on an integrated time-triggered architecture published by Springer-Verlag, USA. He is the author of several journal articles and conference publications. He has also participated in numerous EU research projects (e.g., SAFE POWER, universal, DECOS, and NextTTA). He was the coordinator of the European research projects DREAMS, GENESYS, and ACROSS. His research work focuses on system architectures for distributed embedded real-time systems.