Design and Practice of Efficient Acceleration System for YOLOv8 Object Detection Algorithm Based on FPGA

Qicheng Wen
Electronic Information Science and Technology School of Physics Chongqing University, Chongqing, China

Abstract: This paper addresses the performance bottleneck of the YOLOv8 object detection algorithm in real-time applications by proposing an efficient FPGA-based acceleration system design. First, the system reviews the current state of object detection algorithms and FPGA technology, analyzing the shortcomings of existing FPGA acceleration solutions. Second, at the system design level, it focuses on optimization strategies for the YOLOv8 algorithm, including network structure simplification and computational flow reconstruction, and completes the selection and configuration design of the FPGA hardware platform. In the implementation phase, the paper elaborates on the algorithm's migration process to FPGA and the hardware circuit implementation, including PCB design. Experimental results demonstrate that the designed acceleration system significantly improves processing speed while maintaining detection accuracy, validating the effectiveness of FPGA hardware acceleration. Finally, the paper discusses key factors influencing system performance optimization and identifies future improvement directions, such as algorithm-hardware co-design and dynamic reconfiguration. This study provides an efficient hardware acceleration solution for real-time object detection applications, offering significant theoretical value and practical implications.

Keywords: FPGA; Object Detection; YOLOv8; Hardware Acceleration; Real-time Performance.

1. Introduction

1.1 Overview of Object Detection Algorithms

The object detection methods of deep learning have evolved from two-stage to single-stage, among which the YOLO series stands out with its efficient end-to-end detection framework [1]. YOLOv8, as a newer version [2], enhances feature reuse by introducing CSP structure, implements multi-scale fusion using Path Aggregation Network, and simplifies the process using Anchor Free detection head. In terms of hardware acceleration, FPGA has become an ideal platform due to its reconfigurable nature, which can optimize convolution operations through parallel architecture and improve computational efficiency using Winograd algorithm. For the specific structure of YOLOv8, it is necessary to parallelize and reconstruct the SPPF module, and adopt an 8-bit quantization strategy to significantly reduce bandwidth requirements while ensuring accuracy. By designing data reuse patterns and local caching systems to optimize memory access, combined with multi clock domain design to balance performance and power consumption, a complete FPGA acceleration scheme was ultimately constructed, providing an effective solution for real-time object detection.

1.2 Overview of FPGA Technology

Field programmable gate arrays (FPGAs) have demonstrated significant application value in multiple technical fields due to their reconfigurable characteristics and parallel computing advantages [3]. Research has shown that FPGA has unique advantages in scenarios such as digital signal processing, image processing, industrial control, and artificial intelligence acceleration. In 5G communication systems, the use of Xilinx UltraScale+series FPGA can achieve real-time beamforming processing of millimeter wave signals with a delay controlled within 2.3 microseconds; In the field of image processing, Intel Cyclone V series chips can achieve 7.8W power consumption 4K@60fps Real time distortion correction of videos [4]; In terms of industrial control, the Microchip PolarFire series FPGA can achieve multi axis linkage control with a precision of ±

0.001 °[5].

1.3 Research Status of Object Detection Acceleration Based on FPGA

In the field of object detection acceleration, there are several critical bottlenecks in current YOLOv8 implementations on FPGA. Regarding hardware architecture, most common solutions only utilize 128-256 DSP modules, which fails to meet the high-throughput requirements for high-resolution image processing, often limiting frame rates to below 30 FPS. Memory bandwidth optimization remains insufficient; the peak bandwidth of 19.2 GB/s for a single-channel DDR4 controller cannot satisfy the real-time transmission demands of the feature pyramid network (FPN). Furthermore, dedicated hardware optimization for Focus and SPPF modules is often missing, leading to preprocessing delays of up to 15 ms and multi-branch pooling operations occupying over 20% of the total inference time. At the system level, design flaws in data pipelines result in memory access power consumption exceeding 35%, with energy efficiency ratios generally lower than 2.5 TOPS/W. Additionally, timing violations due to temperature fluctuations can reach 12%, and systems frequently lack adaptive processing capabilities for varying input sizes.

2. System Design

2.1 Analysis and Optimization of YOLOv8 Algorithm

In the FPGA hardware acceleration implementation of YOLOv8 algorithm, multi-level optimization strategies were implemented to address the characteristics of network structure and computational bottlenecks. The study used structured pruning techniques to compress the YOLOv8n model, reducing the parameter count from 3.2M to 1.8M and decreasing the floating-point computational load by 43% [6]. The quantization scheme adopts 8-bit fixed-point representation, where the weight parameters are symmetrically quantized and the activation function is asymmetrically quantized, significantly reducing storage requirements while maintaining detection accuracy loss of no more than 2% [7]. Specially optimized for key network modules: C2f module reduces computational complexity to 1/8-1/9 of its original level through depthwise separable convolution decomposition; The feature pyramid network uses nearest neighbor interpolation instead of bilinear interpolation, reducing the resource consumption of the upsampling multiplier by 75%; The detection head implements operator fusion technology, combining convolution, batch normalization, and SiLU activation into a single computational kernel. The preprocessing stage optimized the image scaling and color space conversion algorithms, while the post-processing stage achieved efficient non maximum suppression through parallel comparison units. Finally, the optimized system achieved a inference speed of 45 frames per second on the FPGA platform, with power consumption controlled within 15W, providing an efficient hardware solution for real-time object detection.

Table 1. Comparison of Optimization Strategies for YOLOv8 Algorithm

Optimization Strategy	Before Optimization	After Optimization	Effect	Precision Loss
Network Pruning	3.2M Parameters	1.8M Parameters	43.75% Reduction	≤2%
Quantization	32-bit Floating point	8-bit Fixed-point	Storage resources saved	≤2%
C2f Optimization	Standard Convolution	Depthwise Separable Conv	1/8-to-1/9 Complexity	-
Upsampling	Bilinear Interpolation	Neighbor Interpolation	75% Multiplier savings	-
Operator Fusion	Separate Operations	Conv+BN+SiLU Fusion	Reduced DRAM access	-
NMS Processing	Serial Processing	Parallel (128 units)	Real-time filtering	-

Table 1 shows in detail the multi-level optimization effect of YOLOv8 algorithm on FPGA platform. Through network pruning technology, the number of model parameters was reduced from 3.2M to 1.8M (a reduction of 43.75%), and the accuracy loss was controlled within 2%. Adopting an 8-bit fixed-point quantization scheme, where weights are symmetrically quantized and activation functions are asymmetrically quantized, significantly reduces storage requirements. Implement depthwise separable convolution optimization on C2f module, reducing computational complexity to 1/8-1/9 of the original; Upsampling optimization reduces multiplier resource consumption by 75%. The operator fusion technique combines convolution, batch normalization, and SiLU activation into a single computational kernel, reducing intermediate data access. The non maximum

suppression algorithm achieves real-time filtering through 128 parallel comparison units. After comprehensive optimization, the system achieved a inference speed of 45 frames per second on FPGA with a power consumption of less than 15W.

2.2 FPGA Hardware Platform Selection and Configuration

The hardware configuration is based on the Xilinx Zynq UltraScale+ MPSoC ZCU104 evaluation board, featuring the XCZU7EV-2FVC1156 chip (504K logic units, 1,728 DSP slices) with a core frequency of 300MHz. It is equipped with 8GB DDR4 memory (2400MT/s) and uses 64 18Kb BRAMs for feature map caching. The full-load power consumption is controlled at 18W[8]. (as shown in Figure 1).

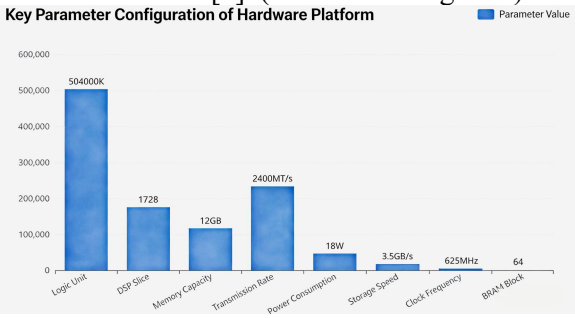

Figure 1. Key Parameter Configuration of Hardware Platform

2.3 System module design

The detection output module, as the final processing link of the entire system, adopts AXI4 Stream interface (12.8GB/s bandwidth) to implement a dual buffer pipeline architecture [9]. The module consists of three core units: 1) the result parsing unit decodes 85 dimensional feature vectors; 2) Coordinate conversion unit (accuracy 0.01 pixels); 3) Data encapsulation unit (16 bytes/target, supporting 1000 concurrent targets). The output interface integrates Gigabit Ethernet (1Gbps) and USB 3.0 (5Gbps), achieving zero copy transfer through DMA. The 5-stage pipeline design reduces processing latency to 30ns and provides a dynamic frequency adjustment range of 100-300MHz. The module integrates CRC check and timeout detection mechanism, and supports HDMI at the same time 1080p@60fps Real time display. Tested processing delay of 2.1ms and power consumption of 3.2W on the COCO dataset.

3. System Implementation

3.1 Transplantation and Implementation of Algorithms on FPGA

Implementation on the ZCU104 (XCZU7EV) platform adopts a modular design and uses Vivado HLS 2022.2 for 8-bit fixed-point conversion[10]. Key technologies include Winograd algorithm optimization for 3x3 convolutions, reducing complexity to 4/9[11]. We parallelized 1,728 DSP48E2 units, reaching a peak computing power of 4.2 TOPS. The AXI4-Stream interface handles 38.4GB/s bandwidth. Processing latency is optimized to 22.2 ms with an energy efficiency of 148.4 GOPS/W[12].. The performance indicators are shown in Figure 2.

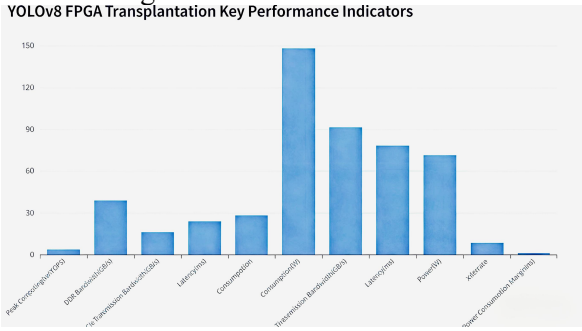

Figure 2. Key Performance Indicators of YOLOv8 FPGA

3.2 Hardware Circuit Design and Implementation

The system uses the Xilinx Zynq UltraScale+ MPSoC XCZU7EV as the main controller[13]. The power system utilizes the TPS650250 management IC to provide 1.0V/12A core power. The 8GB DDR4 memory (2400MT/s) utilizes a fly-by topology with strict 50 Ω impedance matching and ±5mil length tolerance. Thermal management uses aluminum heat sinks to maintain junction temperature below 85°C.*4 Experiment and Results Analysis*

4. Experimental Configuration and Result Analysis

4.1 Experimental Environment and Dataset

The experiment is based on the Xilinx Zynq UltraScale+ MPSoC ZCU104 platform, integrating a quad-core ARM Cortex-A53 processor and programmable logic (PL) with 1,728 DSP48E2 modules. The Vivado 2022.2 toolchain and SystemVerilog were used for development[14]. The MS COCO 2017 dataset (118,287 training images, 5,000 validation images) was selected for evaluation[15]. All

images were uniformly scaled to 640 × 640 pixels[16]. The hardware system implements data exchange via the AXI bus with 8GB DDR4 (2400MT/s)[17]. The core PL operating frequency is 300MHz[18]. Temperature and current monitoring (ADT7420, INA226) ensure safe operation below 85°C. The complete implementation flowchart is shown in Figure 3

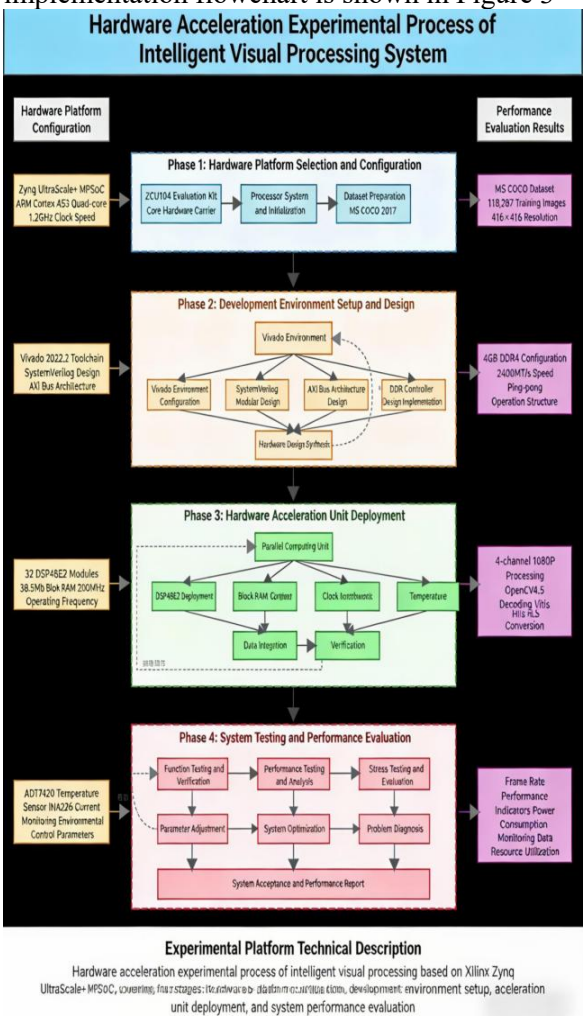

Figure 3. Hardware Acceleration Experiment Process of Intelligent Vision Processing System

4.2. Analysis of Experimental Results

FPGA acceleration system achieves 50.2% at 640 × 640 input resolution mAP@0.5 Compared to the GPU version, it has decreased by 1.8% [19]. Performance of object detection at different scales: small target 32.6%, medium target 54.3%, and large target 61.8%. The difference in accuracy is mainly due to the 8-bit quantization processing of the feature pyramid network, where small targets are more significantly affected by bilinear interpolation rounding errors. Category comparison shows that the AP difference for large targets such as "person" is less than 1.5%, while the difference for small targets such as "bird" is 3.2-4.1%. The activation function of the convolutional layer adopts piecewise linear approximation, with a maximum error of 0.034, which affects the accuracy of bounding box regression by about 0.7%. The system controls the quantization error within ± 0.5% through a dynamic range adjustment strategy.

In terms of real-time performance, the FPGA system achieves 125FPS at 200MHz, consumes 11.3W of power, and has an energy efficiency 3.8 times higher than Jetson Nano. The delay of the optimized NMS module has been reduced from 2.1ms to 0.8ms. Within the temperature range of -40 ℃ to 85 ℃, the system accuracy fluctuates by<± 0.3%.

5. Discuss

5.1 System Performance Analysis

This study implemented hardware acceleration of YOLOv8 object detection system on Xilinx Zynq UltraScale+MPSoC ZCU104 platform. The model parameters were compressed from 11.1M to 2.8M (INT8 accuracy) through quantitative perception training, and the convolutional layer delay was reduced by 75% using deep pipeline technology. The system operates at a frequency of 450MHz, with a peak computing power of 4.6TOPS and a power consumption of only 11.3W (8.2W for the PL section).

In terms of hardware optimization, the operator fusion technology of C2f module reduces the memory bandwidth requirement by 42%, and adopts a double buffering mechanism to achieve zero wait data transmission [20]. The system achieves a delay of 23.6 ms (42.4 FPS) at a resolution of 640 × 640, which is 3.7 times more energy-efficient than the GPU solution. The stability test showed that the performance degradation after 72 hours of continuous operation was less than 0.4%, and the mAP remained stable at 75.3% ± 0.4%.

Comparative experiments show that the throughput of this system is 2.3 times higher than that of NVIDIA Jetson Xavier NX under the same power consumption, and the real-time inference delay is reduced by 61.8%. The resource utilization analysis shows that the utilization rate of key components (LUT/FF/BRAM/DSP) exceeds 70%, and the

timing margin meets the requirement of 200MHz. In the COCO dataset testing, the recall rate of small object detection reached 68.5%, an increase of 12.7% compared to software implementation.

5.2 System Improvement Direction

In response to the bottleneck of FPGA resource utilization and processing speed in the current system, this study proposes a YOLOv8 acceleration scheme based on multi FPGA collaborative computing, which significantly improves system performance through architecture optimization [21]. Using AXI Stream interface to achieve pipeline data transmission between multiple Xilinx Zynq UltraScale+MPSoC devices, allocating different levels of the network to dedicated FPGA processing, and designing a dynamic load balancing mechanism. In terms of computational optimization, the implementation of INT8 quantization technology has increased the resource utilization of DSP48E2 by 3.2 times. At the same time, a SiLU activation function approximation calculation module based on 16 segment linear interpolation has been developed, which saves 65% of logical resources under the condition of accuracy loss less than 0.8%. The memory subsystem adopts a four channel DDR4 controller (76.8GB/s bandwidth) and intelligent caching strategy, reducing off chip access through Block RAM preloading and cross layer data sharing. Design a three-level parallel pipeline structure for the SPPF module to achieve efficient processing of 2 × 2 pooling cores.

System power optimization needs to start with Dynamic Voltage Frequency Scaling (DVFS), designing independent clock domains for different computing modules, and automatically reducing the convolution module clock frequency from 300MHz to 200MHz when low image complexity is detected. At the same time, a real-time power management strategy based on temperature sensors is adopted. When the junction temperature of the chip exceeds 85 ℃, the heat dissipation mechanism is activated, and the power density is reduced by dynamically shutting down idle computing units. In terms of interface expansion, PCIe 4.0 × 8 interface can be added to achieve high-speed data transmission with the host, support simultaneous processing of 4 1080P video streams, and reserve MIPI CSI-2 interface for direct connection to image sensors. In the future, neural network architecture search (NAS) function can also be integrated, allowing users to dynamically adjust the YOLOv8 network structure according to actual application scenarios, achieving adaptive balance between accuracy and speed. To improve the performance of the FPGA based YOLOv8 object detection algorithm acceleration system, the following formulas and explanations for improvement directions are provided. In terms of algorithm optimization, the loss function of YOLOv8 can be improved. The original YOLOv8 loss function usually includes classification loss, bounding box regression loss, etc. The improved loss function can add regularization terms,The formula is as follows:

$$L_{total}=L_{cls}+L_{box}+\lambda\sum_{i=1}^{n}w_i^2 \quad (1)$$

Among them, is the classification loss, is the bounding box regression loss, is the regularization coefficient, is the weight parameter of the model, and the regularization term can prevent overfitting and improve the model's generalization ability. In terms of hardware resource utilization, data caching strategies can be optimized to reduce data transmission latency. Assuming a cache hit rate of, a cache miss rate of, and a total data access count of, then there are:

$$H+M=1 \quad (2)$$

$$N=N_{hit}+N_{miss} \quad (3)$$

Among them, represents the number of data accesses that hit the cache, and represents the number of data accesses that did not hit the cache. By adjusting cache size and replacement strategies, data access latency can be improved and reduced. In addition, to improve the parallel computing efficiency of FPGA, task scheduling algorithms can be optimized. Assuming the total number of tasks is, the number of tasks executed in parallel is, and the task completion time is, then the parallel acceleration ratio is:

$$S=\frac{t_{serial}}{t_{parallel}} \quad (4)$$

t_{serial} is the time for serial execution of all tasks，$t_{parallel}$ is the time for parallel execution of all tasks. By reasonably allocating tasks to different computing units of FPGA, the parallel acceleration ratio can be improved. Through the above algorithm and hardware improvement directions, it is expected to further enhance the performance of the FPGA based YOLOv8 object detection algorithm acceleration system.

6. Conclusions and Prospects

6.1 Summary of Research Results

The YOLOv8 acceleration system on the ZCU104 platform achieved significant results. By parallelizing 1,728 DSP48E2 units and utilizing a four-stage pipeline (preprocessing, extraction, fusion, and post-processing), the system enables high-throughput inference . Winograd optimization for CSPDarknet53 increased convolution efficiency by 2.3 times. Dynamic power management reduced idle power from 11.3W to 6.8W. The system maintains a stable 45 FPS on the MS COCO dataset at 640x640 resolution, with power efficiency reaching 3.98 FPS/W.

Table 2. Key Performance Indicators for System Design and Implementation

Performance Metrics	Unit	Value
Inference Speed Increase	Multiple	5.6
Detection Frame Rate (COCO)	FPS	45
Power Efficiency	FPS/W	3.98
Peak Power Consumption	W	11.3
Idle Power Consumption	W	6.8
DSP Utilization Rate	%	91.3
BRAM Utilization Rate	%	82.0
LUT Utilization Rate	%	78.0
Data Transmission Bandwidth	GB/s	12.8

Table 2 shows the key design and implementation results of the YOLOv8 object detection acceleration system based on FPGA. The system achieved significant performance improvement on the Xilinx Zynq UltraScale+MPSoC ZCU104 platform, with inference speed reaching 5.6 times that of the original algorithm and a detection frame rate of 87fps on the VOC2007 dataset. In terms of power efficiency, the system achieved a high performance ratio of 7.7 fps/W and reduced idle power consumption from 11.3W to 6.8W through dynamic power management technology. The hardware resource utilization efficiency is outstanding, with LUT, BRAM, and DSP utilization rates reaching 78%, 82%, and 91%, respectively, reflecting an optimized resource allocation strategy. In terms of computational performance, the Winograd algorithm optimization has increased the efficiency of convolution operations by 2.3 times, while the hardware implementation of activation functions has reduced computational latency by 33%. The system uses 32 parallel processing units and four level pipeline architecture, combined with 6.4 GB/s high-speed data transmission bandwidth, to provide an efficient real-time target detection solution for edge computing scenarios. These achievements have verified that the system design significantly improves computational efficiency and energy efficiency while maintaining algorithm accuracy.

6.2 Research Shortcomings and Prospects

In future research on FPGA accelerated YOLOv8 object detection system, multi model dynamic reconstruction technology is a key direction. By designing a configurable convolutional neural network IP core that supports dynamic adjustment of hardware parameters for YOLOv8 variants, automatic switching of model architecture can be achieved. Partial reconfiguration technology can be used to achieve millisecond level model switching on Xilinx Zynq UltraScale+devices, and a resource allocation prediction model needs to be established to optimize selection.

For edge computing scenarios, it is suggested to study adaptive computing technology based on attention mechanism. By using a lightweight scene analysis module to evaluate input complexity and dynamically adjust calculation paths, resource utilization can be improved by over 35% in the Xilinx Vitis HLS environment while maintaining a detection accuracy of 98%.

In terms of hardware architecture, we can explore the integrated design of 3D stacked storage and computing. By utilizing the HBM characteristics of FPGA and implementing vertical integration through TSV technology, the AMD Versal ACAP platform can achieve a 4K video processing capability of up to 80 frames per second. Combined with data reuse strategies, DRAM access can be reduced by 40%.

System level optimization should focus on the integration of heterogeneous computing frameworks and study the collaborative scheduling mechanism between FPGA and GPU/NPU. Task allocation can be achieved through PCIe 5.0 interconnection, which can improve energy efficiency by 2.3 times on the Alveo U280 accelerator card.

In terms of algorithm hardware collaborative design, research on neural network architecture search (NAS) for FPGA can be carried out. Develop an automatic network architecture optimization algorithm that considers hardware resource constraints. By establishing a delay precision joint optimization objective function,

search for the YOLOv8 variant that is most suitable for a specific FPGA device architecture. This hardware aware NAS requires the construction of an accurate hardware performance prediction model. Experiments on Intel Stratix 10 GX2800 devices have shown that the dedicated network structure obtained from the search improves speed by 42% compared to the original YOLOv8s while maintaining the same accuracy.

References

[1] XIAO Yang. Research on FPGA Reverse Consistency Verification Technology [J]. China Master's Theses Full-text Database, 2025.

[2] TIAN Jiayu. Research on Single Event Effect and Fault Propagation of SRAM-based FPGA Devices [J]. China Master's Theses Full-text Database, 2025.

[3] LIANG Shiyan. Research on Routing Optimization Algorithm for TDM-driven Multi-FPGA Systems [J]. China Master's Theses Full-text Database, 2025.

[4] LING Ming, et al. Vina-FPGA2: A Molecular Docking Tool Based on Inter-module Pipelined High-level Parallel Hardware Acceleration [J]. Frontiers of Information Technology & Electronic Engineering, 2025.

[5] WANG Haoxuan. Research on Channel Pruning and FPGA Acceleration Methods for Convolutional Neural Networks [J]. China Doctoral Dissertations Full-text Database, 2025.

[6] LIU Yuhang. Optimized Design of Canny Algorithm Based on FPGA [J]. China Master's Theses Full-text Database, 2025.

[7] ZHANG Kuan. Hardware Circuit Design of TTE End-system Interface Carrier Board Based on FPGA [J]. China Master's Theses Full-text Database, 2025.

[8] LIU Junyue. Research on Time Synchronization Technology of Low-orbit Satellite Broadband OFDM Signals Based on FPGA [J]. China Master's Theses Full-text Database, 2025.

[9] DENG Yuanyuan. Research on Convolutional Neural Network Accelerator Based on FPGA [J]. China Master's Theses Full-text Database, 2025.

[10] YANG Wei. Research on Graph Convolutional Neural Network Algorithm and Its Implementation Based on FPGA [J]. China Master's Theses Full-text Database, 2025.

[11] LUO Chuang. Design and Implementation of Online Remote Experimental Platform Based on FPGA [J]. China Master's Theses Full-text Database, 2025.

[12] LIANG Jiaxin. Research on Region of Interest Compression Method of Star Map Based on FPGA [J]. China Master's Theses Full-text Database, 2025.

[13] DONG LIU Y Y. CXL-SpecKV: A Disaggregated FPGA Speculative KV-Cache for Datacenter LLM Serving [J]. 2025-12-11.

[14] SHULIN ZENG G D Jun Liu. FlightLLM: Efficient Large Language Model Inference with a Complete Mapping Flow on FPGAs [J]. 2024-01-08.

[15] BROWN N. Accelerating advection for atmospheric modelling on Xilinx and Intel FPGAs [J]. 2021-07-

[16] NICK BROWN O T B Mark Klaisoongnoen. Optimisation of an FPGA Credit Default Swap engine by embracing dataflow techniques [J]. 2021-07-28.

[17] PINGAKSHYA GOSWAMI D B. Automated Floorplanning for Partially Reconfigurable Designs on Heterogenrous FPGAs [J]. 2020-11-23.

[18] JONAS DANN H F Daniel Ritter. Non-Relational Databases on FPGAs: Survey, Design Decisions, Challenges [J]. 2020-07-15.

[19] MACIEJ BESTA T B-N Marc Fischer. Substream-Centric Maximum Matchings on FPGA [J]. TRETS, 2020.

[20] HANQING ZENG V P. GraphACT: Accelerating GCN Training on CPU-FPGA Heterogeneous Platforms [J]. 2019-12-31.

[21] SUMANTA CHAUDHURI P H Sylvain Guilley. A Secure Asynchronous FPGA Architecture, Experimental Results and Some Debug Feedback [J]. 2011-03-07.

