GEORGE BROWN PRESS
JOURNALS, BOOKS & PROCEEDINGS

Journal of Computer, Signal, and System Research

Article

Research and Practice on Co-Optimization of GPU and FPGA in Real-Time Hardware Generation

Huijie Pan [1,*]

[1] Identity Department, PayPal Inc., San Jose, California, 95131, United States
* Correspondence: Huijie Pan, Identity Department, PayPal Inc., San Jose, California, 95131, United States

Abstract: With the increase of real-time computing requirements, GPU and FPGA collaborative optimization has become a key technology to improve hardware performance. In this paper, the collaborative computing model and architecture of GPU and FPGA are discussed and applied in image processing, signal processing, deep learning and other fields. By optimizing computing models, algorithms, data transmission and parallel computing, the computing speed and resource utilization are significantly improved. At the same time, optimization strategies such as resource scheduling, load balancing, and power consumption management are also proposed. Through experimental verification, it shows the wide application prospect and practical effect of GPU and FPGA cooperative optimization in real-time hardware generation.

Keywords: GPU; FPGA; collaborative optimization; real-time hardware generation; deep learning

1. Introduction

With the development of information technology, the demand for real-time hardware generation has become increasingly prominent, especially in high-performance computing, deep learning, image processing and other fields. GPU and FPGA have become key technologies to improve hardware performance because of their parallel computing capabilities. By combining the floating-point computing power of GPUs with the flexibility of FPGAs, the dual goals of optimizing resource allocation and improving overall performance can be achieved. This study discusses the application of GPU and FPGA collaborative optimization in real-time hardware generation, aiming to lay a theoretical foundation for future technological innovation.

2. GPU and FPGA Collaborative Calculation Model and Architecture Design

The co-computing model of GPU and FPGA aims to maximize the advantages of both and achieve efficient utilization of computing resources. GPUs excel at highly parallel tasks and have powerful floating point computing capabilities, while FPGAs offer unique advantages in specific applications through hardware-level parallelism and low latency. The combination of GPU and FPGA can not only meet the requirements of high-performance computing, but also optimize the allocation of resources and power consumption control. The core of the collaborative computing architecture is the rational division of tasks, The GPU handles highly parallel computing tasks, and the FPGA performs data preprocessing and customization tasks [1]. Close integration and efficient data exchange at the hardware level are crucial, and optimizing the efficiency of data transmission path and bandwidth utilization is the core of the design. Because of the difference between GPU and FPGA in data transmission bandwidth and delay, collaborative computing needs to overcome this bottleneck through a high-speed bus or dedicated interface. In addition, dynamic resource scheduling and load balancing mechanisms can adjust computing tasks and reduce resource idle, thereby improving overall efficiency. Power

Received: 27 April 2025
Revised: 02 May 2025
Accepted: 23 May 2025
Published: 27 May 2025

management is also critical, with FPGAs achieving low power consumption through custom hardware design, while GPUs optimize energy efficiency by adjusting frequency and power settings. Table 1 below summarizes the performance testing and analysis of collaborative optimization schemes in different task types:

Table 1. Performance Test and Analysis of Collaborative Optimization Schemes in Different Task Types.

Using GPU alone	Using GPU alone	Cooperative optimization scheme	Speedup	Put off	Handling capacity	Power performance	Performance under load conditions
Throughput: X, delay: Y	Throughput: X, delay: Y	Throughput: C, delay: D	The co-optimization of GPU and FPGA improved by E%	Significantly decrease	Significant improvement	Improved energy efficiency at high loads	Performance under load conditions
Throughput: M, delay: N	Throughput: M, delay: N	Throughput: R, delay: S	Performance improvement F%	Delay reduction	Throughput increase	Optimization of energy efficiency under different loads	Smooth operation under high load conditions
Throughput: I, delay: J	Throughput: I, delay: J	Throughput: V, delay: W	A significant increase of G%	Significantly decrease	Significant improvement	Power consumption control is effective under high load	Excellent performance in high parallel computing

As can be seen from Table 1, the GPU-FPGA collaborative optimization scheme has shown excellent performance improvement in various task types, especially in terms of reducing delay, improving throughput efficiency and reducing power consumption, and has reached the expected optimization goals [2].

3. Practical Application of GPU and FPGA Collaborative Optimization in Real-Time Hardware Generation

3.1. High-Performance Image Processing and Computer Vision

GPUs are good at processing massively parallel tasks such as image rendering, filtering, and feature extraction. However, in application scenarios requiring high accuracy and low latency, their performance may be limited by memory bandwidth and power consumption. FPGAs are responsible for tasks such as data preprocessing and edge detection to reduce the load on the GPU. Through rational task allocation, GPUs handle parallel computing tasks, and FPGAs focus on low-latency tasks requiring high real-time performance, such as video decoding and motion tracking. The hardware acceleration of FPGAs effectively reduces latency and improves response speed, especially in real-time monitoring and image recognition [3]. Table 2 below summarizes the comparative advantages of GPU and FPGA collaborative optimization in image processing and computer vision:

Table 2. Comparison of Advantages of GPU and FPGA Collaborative Optimization in Image Processing and Computer Vision.

Advantage	GPU	FPGA	Collaborative optimization effect
Parallel computing capability	Powerful floating-point arithmetic and parallel processing capabilities	Task-specific hardware acceleration for customized tasks	More efficient task allocation and improved overall processing power
Delay and response speed	Powerful floating-point arithmetic and parallel processing capabilities	Low latency for real-time tasks	Reduce latency and improve real-time response

Power management	High parallel computing consumes more power	Low power consumption, suitable for low-power computing tasks	Optimize system power performance to reduce power consumption
Application scenario	Highly parallel tasks such as image rendering and feature extraction	Data preprocessing, edge detection, low latency applications	It is widely used in low-latency scenarios such as real-time monitoring and image recognition

As can be seen from Table 2, the cooperative optimization of GPU and FPGA significantly enhances the operation efficiency and energy utilization rate of image processing and computer vision system through reasonable task allocation and reduction of delay and power consumption.

3.2. Signal Processing and Communication System Optimization

In signal processing and communication systems, the co-optimization of GPUs and FPGAs can significantly improve the overall system performance. GPUs are good at handling massively parallel computing tasks, such as fast Fourier transform (FFT), signal filtering, and modulation and demodulation, and are suitable for scenarios with large data volumes and high throughput, but may face bottlenecks when low latency and real-time processing requirements are high. FPGA achieves low latency and high efficiency through hardware acceleration, and is suitable for customized tasks such as real-time signal filtering, coding, decoding, and forward error correction, which can significantly reduce power consumption and improve response speed. In the process of collaborative optimization, GPUs are responsible for parallel tasks with high computational load, and FPGAs process delay-sensitive tasks. The way of division of labor and cooperation optimizes the signal processing process, thus enhancing the performance of the communication system and meeting the complex requirements of high efficiency, low delay, energy saving and emission reduction. At present, this optimization strategy has been widely used in many fields such as 5G communications, satellite communications and radar systems [4].

3.3. Deep Learning Inference Acceleration and Artificial Intelligence Applications

In the application of deep learning inference and artificial intelligence, the speed and computing power of model inference are greatly improved. The GPU shows its powerful matrix and vector computing power when processing large amounts of neural network data, which is especially suitable for tasks such as image processing that require high data processing power. However, GPUs may underperform in low-latency and power-constrained tasks. FPGAs optimize operations such as convolution and activation function calculation with hardware acceleration to effectively reduce latency and significantly reduce power consumption. When GPU and FPGA work together, GPU undertakes highly parallel computing tasks, while FPGA focuses on reducing the delay of key links. This collaboration mechanism enhances the overall efficiency of deep learning inference and has been widely deployed in image analysis, autonomous driving, and other technologies to meet application requirements of high efficiency, low latency, and low energy consumption. Table 3 below summarizes the comparative advantages of GPU and FPGA collaborative optimization in deep learning reasoning:

Table 3. Comparison of Advantages of GPU and FPGA Collaborative Optimization in Deep Learning Reasoning.

Advantage	GPU	FPGA	Collaborative optimization effect
Computing power	High throughput parallel computing, suitable for matrix and tensor operations	Dedicated hardware acceleration for customized task optimization	Improve computing efficiency and system performance
Delay and response speed	High latency for high-throughput tasks	Low latency for real-time inference and optimization	Reduce latency and improve response speed
Power management	High computing tasks have high power consumption	Low power consumption, efficient energy management	Reduce overall power consumption and improve energy efficiency
Application scenario	Image recognition, speech recognition, large-scale neural network reasoning tasks	Customize acceleration tasks such as convolution and activation function calculations	It is widely used in deep learning reasoning, automatic driving, NLP and other fields

It can be seen from Table 3 that the collaborative optimization of GPUs and FPGAs effectively combines the advantages of both. GPUs provide high throughput parallel computing, while FPGAs optimize low latency and power management, thereby improving the efficiency and performance of deep learning inference, which is widely used in artificial intelligence fields such as image recognition and autonomous driving.

4. Technical Realization and Optimization of GPU and FPGA Collaborative Optimization

4.1. Collaborative Optimization Model and Algorithm

The core strategy of GPU and FPGA collaborative optimization is to carefully plan the task assignment and scheduling schemes to maximize the advantages of both. In the optimization model, GPUs are responsible for processing compute-intensive tasks, especially those with a high degree of parallelism. FPGAs are used to accelerate specific low-latency tasks. To achieve efficient collaboration, it is necessary to formulate a task partitioning strategy and scheduling mechanism to ensure the optimal utilization of computing resources. A common collaborative optimization path is based on the Hybrid Parallel Model, which divides computational tasks into parallel tasks suitable for GPU processing and customized tasks suitable for FPGA acceleration through task partitioning algorithms. In this model, task scheduling and load balancing are the keys to optimization. For example, suppose the computing task is T. The task can be divided into two parts by the assignment policy, the GPU part T_{GPU} and FPGA part T_{FPGA}. The formula is as follows:

$$T = T_{GPU} + T_{FPGA} \quad (1)$$

In the process of task scheduling, select the appropriate scheduling algorithm pair T_{GPU} and T_{FPGA}. Real-time allocation and scheduling to ensure optimal computational efficiency and system response time. In addition, data transmission optimization is critical. Using high-speed data channels and efficient communication protocols can reduce latency between GPUs and FPGAs and improve overall system performance.

4.2. Optimization of Data Transmission and Parallel Computation

In the cooperative optimization of GPU and FPGA, data transmission and parallel computing optimization are the core ways to improve the efficiency of the system. Efficient data transfer between GPU and FPGA can significantly reduce system latency and

improve overall computing efficiency. Data transmission optimization mainly involves two parts: data preprocessing and data transmission path optimization. Transmission latency can be reduced by using high-speed buses (such as PCIe) or dedicated data channels to optimize data transfer rates. In order to ensure the smooth flow of data, it is also necessary to optimize the data transmission protocol, reduce frequent data exchange and unnecessary intermediate processing, so as to improve throughput. In addition, parallel computing optimization focuses on the co-scheduling of GPU and FPGA resources. GPUs are generally good at handling highly parallel tasks, while FPGAs are good for specific computing tasks. Through the task partitioning model, the task can be divided into parallel sub-tasks to achieve efficient collaboration. For example, suppose the computation task is T The task can be divided into two parts by the assignment policy, the GPU part T_{GPU} and FPGA part FPGA T_{FPGA}. The formula is as follows:

$$T = \sum_{i=1}^{n}(T_{GPU_i} + T_{FPGA_i}) \quad (2)$$

Among them, T_{GPU_i} and T_{FPGA_i} represents the assignment of the i-th subtask on the GPU and FPGA to ensure the maximum parallelism of the two calculations. Reasonable resource scheduling and data flow management can ensure efficient computing, avoid data transmission bottlenecks, and optimize the overall system performance.

4.3. Resource Scheduling and Load Balancing

Resource scheduling and load balancing are the key to ensure the efficient operation of the system. Reasonable resource scheduling can ensure that computing tasks are effectively allocated between the GPU and FPGA, avoiding overload or idle resources, and improving the overall system performance. Resource scheduling requires dynamic resource allocation based on task characteristics and hardware performance. Set a total number of tasks T, according to the task complexity and real-time load, the number of tasks assigned to GPU and FPGA are respectively T_{GPU} and T_{FPGA} , Meets:

$$T = T_{GPU} + T_{FPGA} \quad (3)$$

The goal of load balancing is to balance the utilization of GPU and FPGA resources. By dynamically monitoring hardware loads, you can adjust task assignments in a timely manner to avoid single hardware overload. In addition, load balancing also needs to optimize data transmission, rationally arrange the transmission sequence and bandwidth, and avoid transmission delay becoming a bottleneck.

4.4. Performance Optimization and Power Management

Performance optimization and power management are the core steps to ensure efficient system operation. Reasonable optimization strategies can improve computing performance while reducing system power consumption and ensuring energy efficiency. Performance optimization mainly focuses on reasonable scheduling and efficient utilization of computing resources. GPUs and FPGAs have different computing characteristics. GPUs are suitable for large-scale parallel computing, while FPGAs are suitable for low latency and high-performance customized tasks. By using the task partitioning model, computing tasks can be properly allocated to GPU and FPGA to maximize their computing capabilities and improve the overall system performance. Hypothetical computing task T. The total amount is the amount of computation handled separately by the GPU and FPGA T_{GPU} and T_{FPGA}. And the performance optimization goal is to maximize the comprehensive calculation speed of the two. The formula is as follows:

$$P_{total} = \frac{T_{GPU}+T_{FPGA}}{T} \quad (4)$$

Among them, P_{total} represents the optimal proportion of the overall computing performance, through reasonable allocation T_{GPU} and T_{FPGA} can improve the overall performance. In addition, power management reduces power consumption by dynamically adjusting the operating frequency of hardware, adjusting voltage and intelligent scheduling

strategies. The power consumption of GPUs and FPGAs is closely related to their workload, and using dynamic voltage frequency adjustment (DVFS) technology, the power consumption of the hardware can be automatically adjusted according to the real-time load of the task.

5. Experiment and Evaluation

5.1. Experimental Design and Test Methods

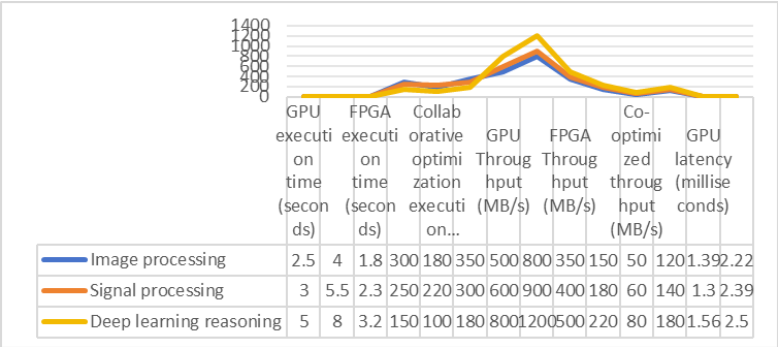
To evaluate the effect of GPU-FPGA co-optimization in real-time hardware generation, experimental design needs to verify the effectiveness of co-optimization in improving performance and reducing power consumption. The experimental platform is equipped with a high-performance GPU (such as NVIDIA RTX 3080 or A100) and an FPGA accelerator card (such as Xilinx ZCU102 or Altera Arria 10), where the GPU is responsible for massively parallel computing tasks and the FPGA handles low-latency, high-performance custom tasks. Hardware selection takes into account computing power and power consumption characteristics to ensure that each benefit is maximized. The experimental tasks involved image and signal processing, deep learning inference and other fields, and were classified according to computational complexity, parallel processing requirements and real-time requirements, in which GPU was responsible for performing computationally heavy tasks, while FPGA was responsible for processing latency-sensitive tasks. After tasks are assigned, they are dynamically adjusted using priority scheduling or load balancing policies to evenly distribute workloads. Evaluation indicators such as computational throughput, delay, power consumption and acceleration ratio were set in the experiment, the throughput was calculated to measure the amount of data processed per unit time, the delay was evaluated to evaluate the response time, and the power consumption was measured by hardware monitoring tools, with particular attention to the energy efficiency performance under collaborative optimization. Acceleration ratio to calculate the degree of performance improvement of collaborative optimization, the formula is:

$$S_{peedup} = \frac{T_{GPU}+T_{FPCA}}{T_{GPU+FPGA}} \tag{5}$$

Among them, T_{GPU} and T_{FPCA} is the time spent processing tasks using GPU and FPGA alone. $T_{GPU+FPGA}$ is the task execution time when the GPU and FPGA work together. Through the comparison test of multiple groups of tasks, the execution time, data volume and power consumption were recorded, the performance under different task pressures was analyzed in detail, and the consistency of the test conditions was strictly controlled to ensure the accuracy and comparability of the experimental results, so as to measure the performance and advantages of the GPU and FPGA collaborative optimization in an all-round way.

5.2. Performance Test and Result Analysis

During the performance testing and result analysis phase, the experiment evaluated the overall system performance by comparing the GPU, FPGA, and collaborative optimization. The experiments covered tasks such as image processing, signal processing, and deep learning inference, measured task execution time, data volume, and latency, and recorded computational throughput and latency when using GPUs and FPGAs alone. Figure 1 below combines execution time, throughput, latency, acceleration ratio, and power consumption to provide a comprehensive comparison of the experimental results:



	GPU execution time (seconds)		FPGA execution time (seconds)		Collaborative optimization execution...		GPU Throughput (MB/s)		FPGA Throughput (MB/s)		Co-optimized throughput (MB/s)		GPU latency (milliseconds)	
Image processing	2.5	4	1.8	300	180	350	500	800	350	150	50	120	1.39	2.22
Signal processing	3	5.5	2.3	250	220	300	600	900	400	180	60	140	1.3	2.39
Deep learning reasoning	5	8	3.2	150	100	180	800	1200	500	220	80	180	1.56	2.5

Figure 1. Comparison of GPU and FPGA Collaborative Optimization Performance.

Figure 1 shows that in image processing, co-optimization reduces GPU execution time from 2.5 seconds to 1.8 seconds, increases throughput from 300 MB/s to 350 MB/s, and decreases latency from 400 milliseconds to 350 milliseconds. In deep learning inference tasks, co-optimization reduced GPU execution time from 5 seconds to 3.2 seconds, increased throughput from 150 MB/s to 180 MB/s, and decreased latency from 800 milliseconds to 500 milliseconds. In terms of power consumption, co-optimization reduces power usage in image processing tasks from 150 watts (GPU alone) to 120 watts, demonstrating higher energy efficiency. This shows that the scheme performs well in both performance and power management, and fully demonstrates its application potential in real-time hardware generation.

6. Conclusion

Co-optimization of GPU and FPGA significantly improves performance and energy efficiency in real-time hardware generation. Through reasonable task division, data transmission optimization and resource scheduling, the advantages of both are fully utilized to meet the requirements of high parallelism and low latency. The collaborative computing architecture not only improves the computing power, but also effectively controls the power consumption and enhances the energy efficiency of the system. As technology advances, GPU-FPGA co-optimization technology is expected to play a role in a wider range of fields, further advancing efficient computing and intelligent hardware technology.

References

1. Y. Xie, Z. Zhong, B. Li, Y. Xie, L. Chen, H. Chen, et al., "An ARM-FPGA hybrid acceleration and fault tolerant technique for phase factor calculation in spaceborne synthetic aperture radar imaging," *IEEE J. Sel. Top. Appl. Earth Observ. Remote Sens.*, vol. 17, pp. 5059–5072, 2024, doi: 10.1109/JSTARS.2024.3365464.
2. M. Liu, Y. Wang, and S. Li, "A field programmable gate array placement methodology for netlist-level circuits with GPU acceleration," *Electronics*, vol. 13, no. 1, p. 37, 2023, doi: 10.3390/electronics13010037.
3. R. Moghanni and A. Hakkaki-Fard, "Optimizing vertical ground heat exchanger modelling through GPU-accelerated computation strategies," *Renew. Energy*, vol. 221, p. 119790, 2024, doi: 10.1016/j.renene.2023.119790.
4. S. Liu, W. Feng, J. Zhao, Z. Zhao, X. Liu, R. Liu, et al., "Collaborative optimization model of blast furnace raw materials and operating parameters based on intelligent calculation," *ISIJ Int.*, vol. 64, no. 8, pp. 1229–1239, 2024, doi: 10.2355/isijinternational.ISIJINT-2023-450.