Domain-Specific Computing Using FPGA Accelerator

● Yasuhiro Watanabe ● Hisanori Fujisawa ● Toshihiro Ozawa

Domain-specific computing is one approach to greatly improving server performance by specially designing a server architecture for a particular application domain. A good way to implement such an approach is to use a field-programmable gate array (FPGA), a commodity device in which processing units and memory blocks can be configured depending on the application requirements. We have developed a domain-specific server for media processing that is specialized for high-speed image retrieval by using an FPGA accelerator. We also developed a design support environment that facilitates FPGA architecture design by enabling the data dependency between modules and a performance bottleneck to be visualized. This technology supports efficient design of high-performance domain-specific FPGA accelerators. In this paper, we describe a partial image retrieval accelerator that serves as a key component of a domain-specific server for media processing and its application to a high-speed image-based document search system. We also describe our design support environment that enables design of high-performance FPGA accelerators.

1. Introduction

The amount of data created worldwide has drastically increased in recent years. In addition, there has been greater introduction of information and communications technology (ICT) and networks, advancement of digital devices such as smartphones, greater deployment of security cameras, and further development of the Internet of Things (IoT). Hence, the amount of image and sensor data will undoubtedly continue to increase. Uncovering hidden values from such data through clever processing is a key to both business and social success. Many applications for processing image and audio data and for machine learning will need to be able to handle an unprecedented amount of data at high speed. Even so, the rate of performance improvement of computers has obviously slowed in the past few years, and increasing computer performance by improving semiconductors in accordance with Moore's Law has nearly reached the limit.

Naturally, the gap between the demands placed on applications—faster and more advanced processing of a large amount of data—and the capabilities of computers with their slow performance improvement continues to grow, requiring a new approach to computer reinforcement. To bridge this gap, we are studying an approach called "domain-specific computing" with which the performance of a computer is drastically improved by focusing on a particular application domain.

In this paper, we introduce the concept of domain-specific computing and then describe a domain-specific server for media processing that finds images on the basis of matches with part of a query image and its acceleration method and application. Then we describe the design support environment for a field-programmable gate array (FPGA), which plays an important role in developing domain-specific servers to be used in various domains in a timely manner. Finally, we discuss the achievements we have made so far and future challenges.

2. Domain-specific computing

Domain-specific computing is an approach that drastically improves server performance and operability by optimizing hardware and software in accordance with the characteristics of the domain specialized for

each application field.

When a general purpose CPU cannot offer sufficient performance, an FPGA and/or a graphics processing unit (GPU) are good options for accelerating processing speed. For a domain requiring faster I/O or energy saving, flash storage and/or nonvolatile memory can be used to improve I/O performance and reduce energy consumption. Offering high performance for each domain's major functions through various devices, optimally configured for the characteristics of a domain, drastically improves performance and operability of the applications belonging to the domain. That is the idea of domain-specific computing. We are developing a domain-specific server to achieve this.

Because of the continuous improvement in general purpose CPU performance in accordance with Moore's Law, it has long been difficult to maintain the excellent features of an accelerator using an FPGA and/or GPU. However, the slowdown in improvements to semiconductor performance has helped accelerators keep their advantage. As FPGAs and GPUs have become more widely used and less expensive, technology to differentiate performance by specifying a domain while using these general devices will become more important.

An FPGA, among other things, can be a useful device for achieving a server specialized for a domain because it can be configured with computing units, data bit width, and memory as appropriate for the content of the data and computing requirements handled by the application. Being imperfect like everything else, an FPGA naturally cannot improve all aspects of processing performance. To achieve high performance, it is necessary to appropriately select applications and architectural designs suitable for the processing characteristics. Having an easy design method is also essential for accelerator development.

Against this backdrop, we have been developing an acceleration technology based on using an FPGA and a design method that facilitates FPGA circuit design.

3. Domain-specific server for media processing

Digital images and audio data, i.e., media data, are believed to account for more than half of all data being created. Media processing to effectively use media data is an important technological area that requires higher processing speed. For instance, many corporations create a large amount of digital documentation containing figures and photos for presentations and other purposes every day. The efficiency with which a document is created can be expected to improve if the images that a person looks for can be quickly found from among an enormous pool of documents.

As the first step toward realizing a domain-specific server, we developed a domain-specific server for media processing.

3.1 Partial image retrieval

Partial image retrieval is a comprehensive search function using an image itself as a query to retrieve similar image(s) that partially match it. It is an effective technology that enables efficient use of figures and photos when creating documents.

Figure 1 shows the flow of the partial image retrieval process that we developed. First, a local feature descriptor, the binary robust independent elementary features (BRIEF)[1] descriptor, is calculated for each feature point assigned to the partial region in the query image and the database images. BRIEF uses the magnitude relationships of randomly selected pixel pairs around the feature point. In our method, 128 bits

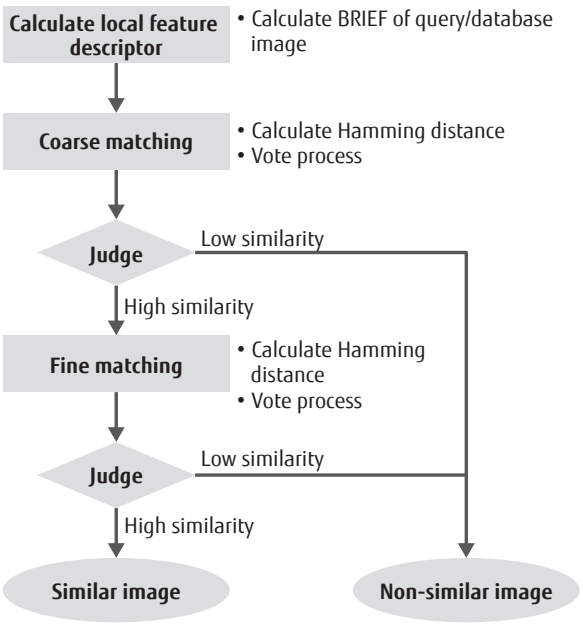

Figure 1
Process flow of partial image retrieval.

extracted from 128 pixel pairs (256 pixels) are defined as the feature descriptor for a single feature point.

Once the feature descriptor is calculated, the similarity between partial regions of images is judged by comparing the BRIEF of each feature point of the query image and the database images. The system finds the feature point of the query image where the Hamming distance to each feature point of the database images is the smallest and identifies corresponding pairs between images. Repeating this procedure for all feature points makes it possible to extract similarities of partial regions.

This comparison of feature points is divided into two steps–coarse and fine matching–so that the amount of calculation can be reduced by terminating the process when the similarity during coarse matching is less than a threshold. Partial image retrieval requires comprehensive comparison of the BRIEF of each feature point between images, and this leads to an enormous amount of calculation that would take conventional general-purpose servers many hours to perform.

3.2 Acceleration with FPGA

We significantly increased the search speed by off-loading the high-load processing of the feature descriptor calculation and matching for partial image retrieval onto an FPGA.

To achieve faster processing using an FPGA, it is essential to have an effective configuration of computing and memory units appropriate for the processing characteristics of the application and to have an optimally designed control flow. The typical operational clock frequency of an FPGA circuit is several hundreds of MHz, about one-tenth that of today's CPUs. The key elements to increase the speed with an FPGA are (1) use small and highly efficient computing units specialized for the application, (2) mount the units in a highly parallel manner in the FPGA resource, and (3) ensure the FPGA has a high working rate.

We were highly successful in achieving these three elements in implementing our partial image retrieval. **Figure 2** shows the overall block diagram of the partial image retrieval accelerator we developed. We mounted 32 feature calculation modules and 6 matching modules in the accelerator. Each matching module

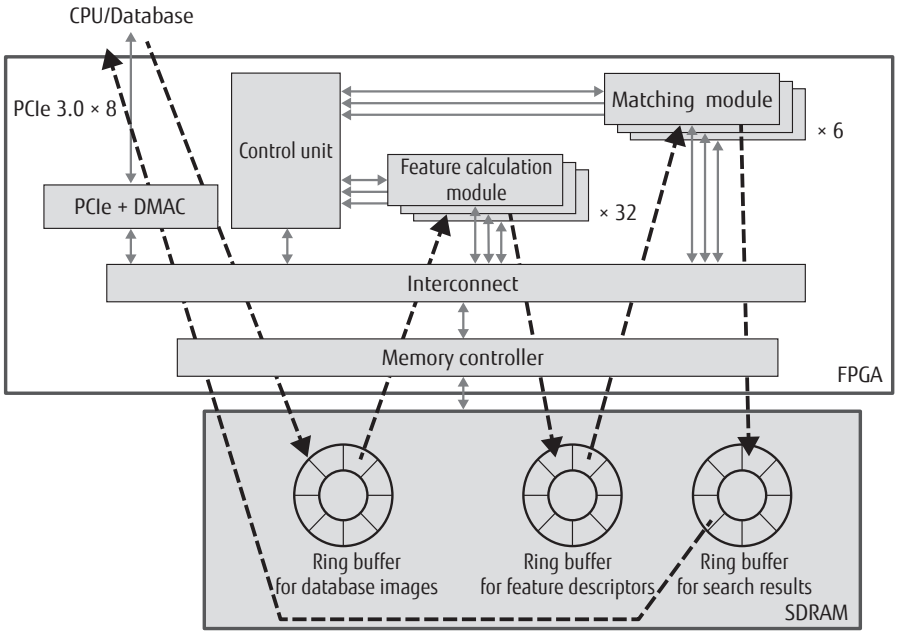

Figure 2
Overall block diagram of partial image retrieval accelerator.

had 64 dedicated computing units that can calculate the Hamming distance and the minimum distance required to identify the pair of feature points at each clock. The computing units were mounted in a highly parallel manner. This configuration comprises one matching module, and six of these modules are used to identify 384 corresponding points for the feature point pairs at each clock.

To achieve a high working rate for the computing modules, we also developed a circuit capable of supplying data to the highly parallel computing units and a scheduling technology that can efficiently use the data read by considering the data flow. As execution of fine matching depends on the result of coarse matching in this algorithm, an ordinary processing flow does not ensure smooth pipelined execution. Our scheduling technology optimizes the processing order so that the data previously read are not wasted even if the subsequent processing target is changed due to the matching result. It thus configures a system with smooth pipelined execution. As a result, the highly parallel computing units run at a high working rate, achieving excellent retrieval performance.

3.3 Performance and application of domain-specific server for media processing

Applying these acceleration technologies to the partial image retrieval accelerator and mounting it on an FPGA, we developed a high-speed partial image retrieval server that achieves a speed more than 50 times higher than that of a general-purpose server.[2] We used Bittware S5-PCIe-HQ (Altera Stratix V GX AB FPGA) as the FPGA board and FUJITSU Server PRIMERGY CX2570 M1 for the server. **Table 1** shows the implementation results including the number of resources that the FPGA circuit uses and other data. A single unit of the high-speed partial image retrieval server that we developed retrieves an image corresponding to the specified area of the query image from more than 10,000 images stored in the database in about one second.

Applying the server to a document retrieval system can reduce the time required for comprehensively searching partial images in documents to seconds from minutes, enabling interactive search using an image. This will contribute to the efficient reuse of a large amount of digital documentation stored in an office.

Table 1
Results of implementation with FPGA.

Number of logic modules (usage rate)	313,816 (87%)
Number of registers (usage rate)	434,916 (30%)
Number of DSP blocks (usage rate)	72 (20%)
Number of block memory bits (usage rate)	43,901,724 (81%)
Clock frequency	200 MHz

DSP: Digital signal processor

4. FPGA design support environment

An FPGA in which circuit reconfiguration is possible has become an important device for domain-specific computing, and its efficient design has become possible thanks to advanced high-level synthesis.[3] Designing a high-performance FPGA circuit requires advanced design skills as highly parallelized computing units must be implemented on limited resources and operated with high efficiency. Architectural-level structure design considering an optimum parallel arrangement of larger processing units and smooth pipeline processing without data stall is even more critical for high-performance design.

One of the important issues to consider when designing an architectural-level structure is designing the performance and structure of each module for pipeline processing. In such processing, the processing speed of the module with the worst performance determines the speed of the entire device, so a module identified as a performance bottleneck needs to be reinforced. While having a parallel arrangement is effective for improving performance, more hardware resources become necessary. The structure should be designed by considering the amount of hardware resources required.

In our efforts to contribute to a structure in which the performance of each module is optimized, we have established a design support environment providing visualized information so that designers can intuitively find a structure helpful for architectural design, including structures to avoid performance bottlenecks. From the input C language code, the results of the analysis of modules that could be a bottleneck are visualized in the environment. The designer seeks a way to modify the module by using the visualized analysis results and sets a performance target. The target can

be directly entered as numerical values to be reflected in the visualized information. Repeating these steps, the designer can optimize overall performance while modifying the performance of the bottleneck module. (**Figure 3**)

The design support environment uses a bar graph with time represented on the X-axis showing initiation interval (**Figure 4**). The initiation interval is the time between the starting point of a process after receiving a set of data and the point when another set of data can be received. The module with the longest interval is the bottleneck and thus determines the performance of the entire throughput; it is emphasized on the screen. For a module with a layer structure, a bottleneck in a lower layer is shown by selecting the module from the list of modules, and the intervals of lower layer modules are shown. Furthermore, the graphical chart indicates the data dependence relationships between modules by showing the next module starts just after the previous module finishing.

Usage of hardware resources needs to be considered in FPGA design, and the usage rate for each resource (FPGA logic elements, internal RAM on FPGA, and circuit dedicated to arithmetic operation [DSP: digital signal processor]) is also shown. Different colors are used when resource usage exceeds a certain level.

The visualized information helps a designer optimize the architectural level. A lack of resources can

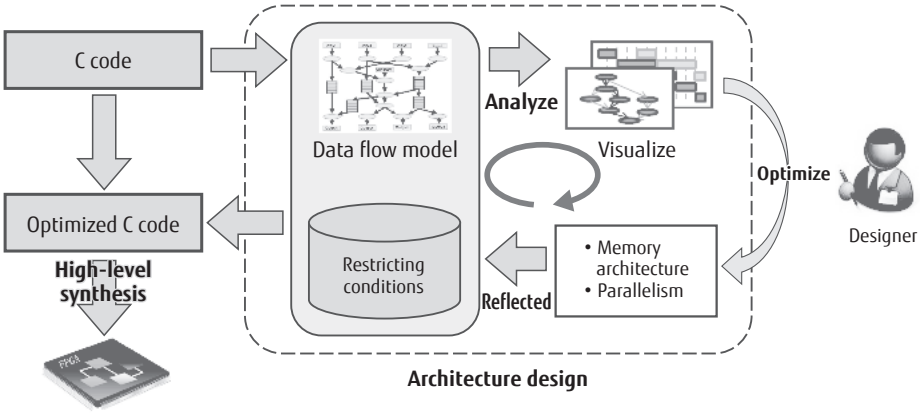

Figure 3
Architecture design flow.

Figure 4
Screen of FPGA design support environment.

be compensated for by decreasing the performance of a module with excessively high performance. The ratio between the initiation interval of the entire circuit and that of each module is shown as the operation ratio. For example, consider a module with an operation ratio of 50%, meaning it works only half the time. Resource usage can be reduced by cutting the number of parallel processings in this module or by integrating the processing of similar modules.

5. Conclusion

In this paper, we introduced the concept of domain-specific computing and its implementation, a domain-specific server for media processing using an FPGA. We also described an FPGA design support environment for developing it.

As the workload continues to increase in many fields, there will be more cases where a general-purpose server does not satisfy performance requirements. Fujitsu Laboratories Ltd. is working to achieve practical application of a domain-specific server to perform high-speed processing of the workload in certain fields along with an FPGA design support environment.

References

1) M. Calonder et al.: Brief: Binary robust independent elementary feature. European Conference on Computer Vision, pp. 778–797 (2010).
2) H. Matsumura et al.: An FPGA-accelerated partial duplicate image retrieval engine for a document search system. IEEE Winter Conference on Applications of Computer Vision (WACV 2016).
3) R. Nane et al.: A Survey and Evaluation of FPGA High-Level Synthesis Tools. IEEE Trans. on CAD of Integrated Circuits and Systems, Vol. 30, No. 10, Oct. pp. 1591–1604 (2016).

Yasuhiro Watanabe
Fujitsu Laboratories Ltd.
Mr. Watanabe is currently engaged in research on domain-specific computing.

Hisanori Fujisawa
Fujitsu Laboratories Ltd.
Mr. Fujisawa is currently engaged in research related to FPGAs.

Toshihiro Ozawa
Fujitsu Laboratories Ltd.
Mr. Ozawa is currently engaged in research on domain-specific computing.

