Heterogeneous Multicore Platform with Accelerator Templates and Its Implementation on an FPGA with Hard-core CPUs

Yasuhiro Takei, Hasitha Muthumala Waidyasooriya, Masanori Hariyama and Michitaka Kameyama
Graduate School of Information Sciences, Tohoku University
Aoba 6-6-05, Aramaki, Aoba, Sendai, Miyagi, 980-8579, Japan
Email: {takei, hasitha, hariyama, kameyama}@ecei.tohoku.ac.jp

Abstract—*Heterogeneous multi-core architectures with CPUs and accelerators attract many attentions since they can achieve power-efficient computing in various areas from low-power embedded processing to high-performance computing. Since the optimal architecture is different from application to application, finding the most suitable accelerator is very important. In this paper, we propose an FPGA-based heterogeneous multi-core platform with custom accelerator templates. Accelerator templates can be reused after optimizing for different applications. According to the evaluation, the proposed platform gives comparable performance to the industrial heterogeneous multicore processors at around 1W of power.*

Keywords: Heterogeneous multicore processor, FPGA, Multimedia processing, High-performance-computing

1. Introduction

Applications used in low-power embedded processing to high performance computing have different tasks such as data-intensive tasks and control-intensive tasks. Therefore, optimal architecture is different from application to application. Heterogeneous multicore processing is proposed to execute applications power-efficiently. It uses different processor cores such as CPU cores and accelerator cores as shown in Fig.1. If the tasks of an application are correctly allocated to the most suitable processor cores, all the cores work together to increase the overall performances.

Examples of low-power heterogeneous multi-core processors are [1] and [2]. The former has multiple cores of CPUs and ALU arrays. The latter has multiple cores of CPUs, a micro-controller and SIMD (single-instruction multiple-data) type processors. Such commercially available processors are partially programmable so that a part of the data path and computations of processing elements (PEs) can be changed to some extent. However, due to the wide variety of tasks and their different memory requirements, this programmability is not enough to extract sufficient performance. Moreover, the programming environments in various heterogeneous architectures. Therefore, each time the architecture changes, large design time is required to re-map the application into the new architecture.

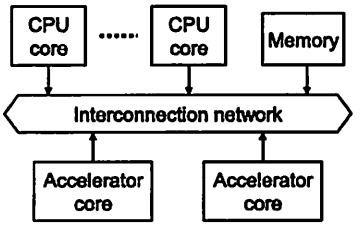


Fig. 1: Heterogeneous multi-core processor architecture

To solve these problems, we propose an FPGA-based platform for heterogeneous multicore processors to explore accelerator architectures suitable for applications. Recently, speed and power consumption of FPGAs are greatly improved, and it would be very practical to use the FPGA-based platform for real applications. The proposed platform consists of CPU cores suitable for control-intensive tasks and custom accelerator cores suitable for data-intensive tasks. The use of the architecture templates reduces the design effort to explore the architectures suitable for applications. It would also make it easy to re-use the same software on different accelerators derived from the same template. Moreover, the high reconfigurability of FPGAs enables to adopt the different types of accelerators for a single application depending on the nature of tasks. The major disadvantage of FPGA-based processors over the commercially available once is the low-performance of CPU cores since CPU cores are generated using look-up tables. Such "soft-core CPUs" cause large computation time and large data transfer time. However, recent FPGAs such as Xilinx Zynq and Altera Cyclone V contain "hard-core CPUs" operating at about 8 times faster than the soft-core CPUs.

This paper is an extension of the work done in [3] which explains the basic idea of the heterogeneous multicore platform. However, the soft-core CPU in [3] is replaced by a low-power hard-core CPU ("Cortex-A9 dual core ARM processor") using Xilinx Zynq so that the processing and data transfer time are significantly reduced. In this paper, as a typical architecture templates, we consider two types of custom accelerators: SIMD one-dimensional PE array (SIMD-1D) and MIMD two-dimensional PE array (MIMD-2D). The SIMD-1D accelerator is suitable for executing simple operations at a high degree of parallelism. The proposed

SIMD-1D accelerator is designed similar to the GPU data path to use the CUDA (compute unified devise architecture) [4] programming language. The MIMD-2D accelerator is suitable for executing complex operation at a medium degree of parallelism. To increase the memory access speed, we introduce a custom hardware called address generation unit (AGU). We can also reconfigure the data path, the number of PEs, the number of memory modules, and memory capacity according to the requirements of a given task to optimize the performance. The evaluation demonstrates that the proposed FPGA-based platform achieves good performance and low-power consumption comparable to industrial heterogeneous processors such as RP1 [1].

2. Heterogeneous multicore platform

2.1 Overall architecture

This section explains the architecture of the heterogeneous multi-core platform. Figure 2 shows the overall architecture. An external DDRII SDRAM is connected to the CPU core through the FPGA board. The custom accelerators have different architectures such as SIMD-1D and MIMD-2D.

It is important to reduce data-transfer time between cores for processing faster in heterogeneous multicore. In previous work [5], window-based image processing time and memory capacity are reduced by using optimal memory allocation and a data-transfer scheme. For further reduction the processing time, we overlap the data-transfer with data processing on different cores as shown in Fig.3. In FPGAs, We can determine the optimal number of accelerator cores and PEs so as to minimize the processing time.

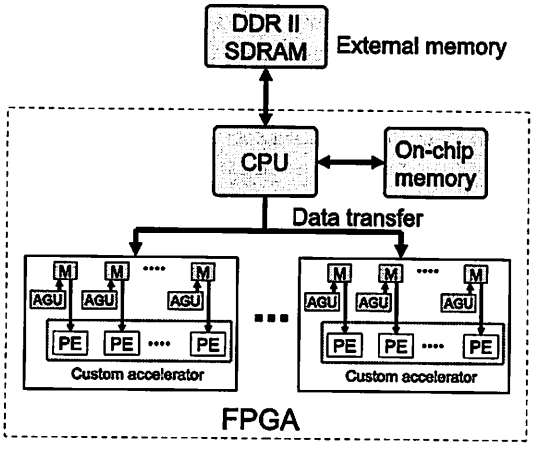


Fig. 2: Proposed heterogeneous multi-core architecture

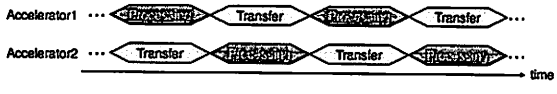


Fig. 3: Overlapping data-transfer and processing

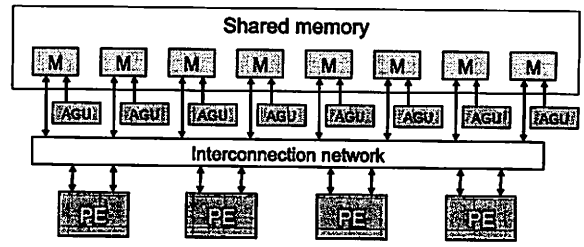


Fig. 4: SIMD-1D architecture

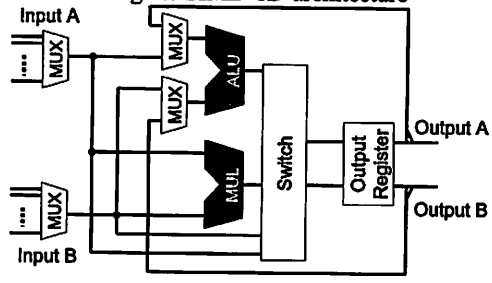


Fig. 5: Architecture of the PE

2.2 SIMD-1D accelerator

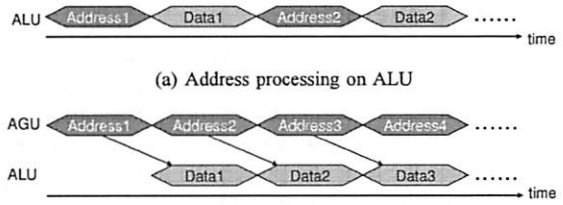
The proposed SIMD-1D accelerator is designed similar to the GPU accelerator so that we can use the same CUDA code. The basic idea of the SIMD-1D accelerator is discussed in [6]. It has a 1-dimensional array of PEs connected to the shared memory as shown in Fig.4. AGUs are included to increase the address generation speed. To execute an application, we have to divide it into independent threads where several of them can be executed in parallel. After the execution is finished, new threads are fed. When all the threads are executed, the resulting data are read by the CPU.

Figure 5 shows the architecture of a PE. It consists of a 16bit fixed-point ALU and a multiplier. Operations such as addition, accumulation subtraction, comparison and absolute difference computation are done in the ALU, and multiplication is done in the multiplier. Multiply-accumulation is done by a pipelining the multiplier and the adder.

In CPUs, the address calculation and data processing are done in the same ALU as shown in Fig.6(a). Therefore, when the addresses are calculated, we cannot do data processing. In the proposed architecture, the address calculation is done in the AGU shown in Fig.6(b). The address calculation and data processing are done in parallel so that we can reduce the total processing time. A detailed description about AGUs is given in [5]. As shown in Fig.2, accelerators in the proposed heterogeneous platform contain AGUs.

2.3 MIMD-2D accelerator

The proposed MIMD-2D accelerator is designed based on the FE-GA accelerator [1] that has a dynamically reconfigurable PE array. Figure 7 shows the proposed MIMD-2D accelerator. It consists of a 2-dimensional array of PEs,



(a) Address processing on ALU

(b) Address processing on AGU

Fig. 6: Address processing

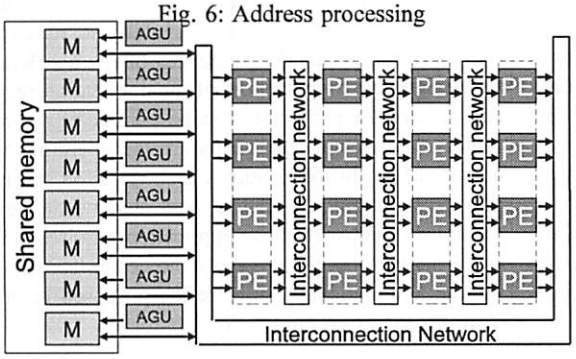


Fig. 7: MIMD-2D architecture model

local memory modules and AGUs. In order to simplify the interconnection network while still meeting the streaming applications, we limit the interconnection network; only leftmost PEs can directly retrieve data from local memory modules, and only rightmost PEs can directly write data to local memory modules. PEs, AGUs and interconnection network are dynamically reconfigurable. To implement applications, we have to divided it into multiple contexts that execute sequentially. Within a context, we can perform parallel computations. The computation starts after the configuration data of multiple contexts are written to the configuration memory of the accelerator. When the computation is finished, the resulting data are read by the CPU.

3. Evaluation

We implement the proposed heterogeneous multicore platform on Xilinx Zynq-7000 EPP ZC702 evaluation kit [7]. Since SIMD-1D and MIMD-2D architectures have different topologies, we perform two comparisons to evaluate the architectures. In the first comparison, the number of look-up-tables (LUTs) in both accelerators is a constant. In the second comparison, the degree of parallelism of the memory access is a constant. As shown in Table 1, SIMD9 and MIMD12 accelerators have almost the same number of LUTs. SIMD4 and MIMD12 accelerators have the same number of memory modules. Therefore, the degree of parallelism of the memory access is the same. In parallel processing, both the number PEs and the degree of parallelism with the memory are equally important.

We compare the processing time of filter computation and SAD-based template matching [8]. The image and window

Table 1: Specification of accelerator cores

Accelerator core	Number of PEs	Number of LUTs	Number of memories	Degree of parallelism
SIMD4	4×1	3301	8 (16kB)	4
SIMD9	9×1	7354	18 (18kB)	9
MIMD12	4×3	7322	8 (16kB)	4

sizes and the operating frequency are 256×16, 16×16 and 100MHz respectively. Table 2 shows the comparison of SIMD-1D (SIMD9) and MIMD-2D (MIMD12) accelerators when the number of LUTs is a constant. For the filter computation, the processing time of the SIMD-1D accelerator is less than half of that of the MIMD-2D accelerator. The SIMD-1D accelerator has a one-dimensional PE array, where all 9 PEs are directly connected to the memory as shown in Fig.4. The MIMD-2D architecture has a two-dimensional PE array of 4×3 where only leftmost 4 PEs can directly retrieve data from the local memory as shown in Fig.7. Therefore, the SIMD-1D accelerator has the higher degree of parallelism of memory access than the MIMD-2D accelerator. In the SAD computation, SIMD-1D accelerator is slightly faster than MIMD-2D accelerator. SAD computation requires two types of operations: absolute difference and addition. the MIMD-2D accelerator can perform these two operations at the same time by pipelining while SIMD-1D accelerator cannot. However, the processing time of the SIMD-1D accelerator is still smaller due to its high degree of parallelism. If we use an application that have three or more types of operations, the MIMD-2D accelerator could give much better results.

Table 2: Comparison 1 : The same number of LUTs

Application	Accelerator core	Processing time (ms)
Filter	SIMD9	0.069
	MIMD12	0.154
SAD	SIMD9	0.139
	MIMD12	0.154

Table 3 shows the comparison of SIMD-1D (SIMD4) and MIMD-2D (MIMD12) accelerators when the degree of parallelism of the memory access is a constant. In the filter computation, the processing times of the SIMD-1D and MIMD-2D accelerators are the same. This is because, multiplication and addition operations are pipelined in both accelerators, so that two operations are performed in one cycle. Moreover, both accelerators have the same degree of parallelism. In the SAD computation, the processing times in MIMD-2D accelerator is about half of that in SIMD-1D accelerator. As described above, the MIMD-2D accelerator can pipeline different type of operations (absolute difference and addition in SAD computation). Hence, MIMD-2D can obtain higher degree of parallelism of operations compared to the SIMD-1D accelerator under the condition of the same number of memory modules.

Let us compare the FPGA-based platform with conventional industrial heterogeneous multicore processors. Figure 8 shows the implemented architecture. There are MIMD-

Table 3: Comparison 2 : The same degree of parallelism

Application	Accelerator core	Processing time (ms)
Filter	SIMD4	0.156
	MIMD12	0.154
SAD	SIMD4	0.318
	MIMD12	0.154

2D accelerator cores which process the filter computation in parallel. Table 4 shows the resource utilization on the FPGA with four MIMD16 cores. Since the FPGA design tool removes unused units on the implemented architecture automatically, the resource utilization is smaller than expected. Note that the number of accelerator cores and the number of PEs in one core can be selected depending on the applications.

Table 5 shows the comparison of the filter computation time for the proposed FPGA-based platform and RP1 [1]. The image size is 640 × 480. The number of PEs on FPGA is 64, and it is equal to using two FE-GAs in RP1. When the number of FE-GA cores is two, the processing time on the proposed platform is very similar to that of RP1. The power consumption of both processors is around 1W. In conclusion, the FPGA-based heterogeneous multicore architecture provides comparable performance to the RP1 heterogeneous multicore processor.

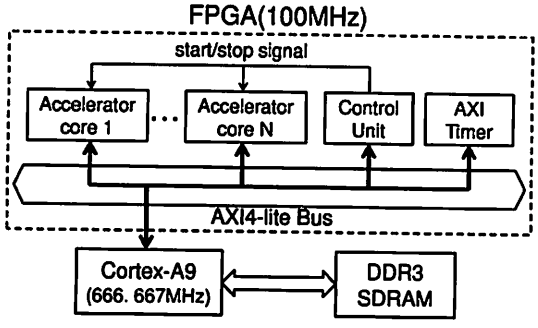


Fig. 8: Implemented architecture

Table 4: Resource utilization of four MIMD16 cores

Module	LUT	Register	Block RAM	DSP
Accelerators	1044	1604	18	16
Control unit	28	28	0	0
AXI timer	312	217	0	0
AXI Interconnect	397	182	0	0
Total	1781(3%)	2031(2%)	18(13%)	16(7%)

Table 5: Comparison of processing time

Window size	Processing time (ms)	
	Zynq 1xCortex-A9(666.667MHz) + FPGA(100MHz)	RP1 [5] 1xSH-4A(600MHz) + 2xFE-GA(300MHz)
12 × 12	46.51	36.24
18 × 18	70.50	72.94
24 × 24	115.89	96.55

4. Conclusion

We have proposed an FPGA-based heterogeneous multicore platform with custom accelerators. The accelerator cores are customizable for each application. Dedicated AGUs are used to increase the processing speed and to reduce the area and power. We evaluate the proposed platform using several examples and show that the proposed platform has performance comparable to industrial heterogeneous processors. To select the best accelerator for a given application, we have to match the requirements of the application with the properties of the accelerator under the design constraints. Most of the application requirements and accelerator properties can be parameterized and represented. The design constraints are the operating frequency, amount of hardware resources such as LUTs and memories, power consumption, etc. Our next step would be to find a relationship between those application requirements and the accelerator properties to satisfy the design constraints. Then we can automatically optimize the proposed heterogeneous platform for given applications.

Acknowledgment

This work is supported by MEXT KAKENHI Grant Number 12020735.

References

[1] H. Shikano, M. Ito, M. Onouchi, T. Todaka, T. Tsunoda, T. Kodama, K. Uchiyama, T. Odaka, T. Kamei, E. Nagahama, M. Kusaoke, Y. Nitta, Y. Wada, K. Kimura, H. Kasahara, "Heterogeneous Multi-Core Architecture That Enables 54x AAC-LC Stereo Encoding", *IEEE Journal of Solid-State Circuits*, Vol.43, No.4, pp.902-910, 2008.

[2] H. Kondo, M. Nakajima, N. Masui, S. Otani, N. Okumura, Y. Takata, T. Nasu, H. Takata, T. Higuchi, M. Sakugawa, H. Fujiwara, K. Ishida, K. Ishimi, S. Kaneko, T. Itoh, M. Sato, O. Yamamoto and K. Arimoto, "Design and Implementation of a Configurable Heterogeneous Multicore SoC With Nine CPUs and Two Matrix Processors", *IEEE Journal of Solid-State Circuits*, Vol.43, No.4, pp.892-901, 2008.

[3] H. M. Waidyasooriya, Y. Takei, M. Hariyama and M. Kameyama, "FPGA implementation of Heterogeneous Multicore Platform with SIMD/MIMD Custom Accelerators", IEEE International Symposium on Circuits and Systems (ISCAS), pp.1339-1342, 2012.

[4] NVIDIA Corporation, "NVIDIA CUDA Programming Guide" Ver2.2.1, 2009.

[5] H. M. Waidyasooriya, Y. Ohbayashi, M. Hariyama and M. Kameyama, "Memory Allocation Exploiting Temporal Locality for Reducing Data-Transfer Bottlenecks in Heterogeneous Multicore Processors", IEEE Transactions on Circuits and Systems for Video Technology, Vol.21, No.10, pp.1453-1466, 2011.

[6] H. M. Waidyasooriya, M. Hariyama and M. Kameyama, "Architecture of an FPGA-Oriented Heterogeneous Multi-core Processor with SIMD-Accelerator Cores", International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA), pp.179-186, 2010.

[7] http://www.xilinx.com/products/boards-and-kits/EK-Z7-ZC702-G.htm

[8] M. Hariyama, H. Sasaki, and M. Kameyama, "Architecture of a stereo matching VLSI processor based on hierarchically parallel memory access", IEICE Trans. Inform. Syst., Vol.E88-D, No.7, pp.1486.1491, 2005.