Reconfigurable Logic Circuits for Intelligent Human Sensing

SHUICHI ICHIKAWA

Dept. Knowledge-based Information Engineering, Toyohashi University of Technology
Toyohashi 441-8580, Japan

Phone: +81-532-44-6897, Fax: +81-532-44-6873

E-mail: ichikawa@tutkie.tut.ac.jp

This paper presents the concept of the reconfigurable preprocessors for intelligent human sensing. Reconfigurable logic can realize high-performance preprocessing of sensor data with flexibility and low power consumption. This paper also summarizes the related works and the future research plan by the author.

1. Introduction

It is well known that the custom circuit for a specific application can accelerate the execution of that application. However, custom circuits were not so widely accepted, because it was difficult to design and manufacture them. Now, recent developments in EDA (Electronic Design Automation) tools have broken the potential barriers to the design of custom circuits. The only problem left is manufacturing them.

A usual LSI device has its function fixed when it is manufactured, and hundreds of the same devices are manufactured together, as is the nature of LSI technology. This means that we have to fabricate hundreds of custom LSI device for a fixed application, which drives up the cost of the device to a prohibitive level.

A reconfigurable logic device is a kind of LSI that can change its logic function (*configuration*) at *run-time*. Using reconfigurable devices, we can implement any custom circuit regardless of quantity. Field Programmable Gate Array (FPGA) is a kind of reconfigurable logic device. The advances in FPGA technology have been truly amazing in these last 10 years, so that today we can purchase an FPGA device with 1—10 million gates at a reasonable price. They are no longer only for researchers but also widely used in the industrial world.

In this study, the author investigates the possibilities of applying reconfigurable logic technology to intelligent human sensing. Figure 1 illustrates the concept of reconfigurable sensor preprocessors.

Huge numbers of sensors are involved in the intelligent human sensing system. These sensors continuously yield enormous amounts of data that must be processed without delay since it is both impossible and futile to record it all. As these data are generated by sensor hardware, their format is relatively simple and fixed. Custom hardware is well suited to process such large amounts of simple data rapidly, whereas it is impractical to handle them by microprocessors that are intrinsically sequential and weak in bulk data transfer. Even the recent high-performance microprocessors cannot escape from the bandwidth limitation between processor and memory (the *von Neumann bottleneck*). For sensor preprocessing, it is natural to adopt a hierarchical design, which includes a central processing unit and low-level preprocessors.

One of the possible problems is that hardware preprocessors are less elastic than software in supporting various applications. However, a dynamically reconfigurable preprocessor is metamorphic in adapting to each application. The design cost would not be a major concern, because sensor outputs are rather simply formatted. Hence, it is relatively easy to implement such custom circuits.

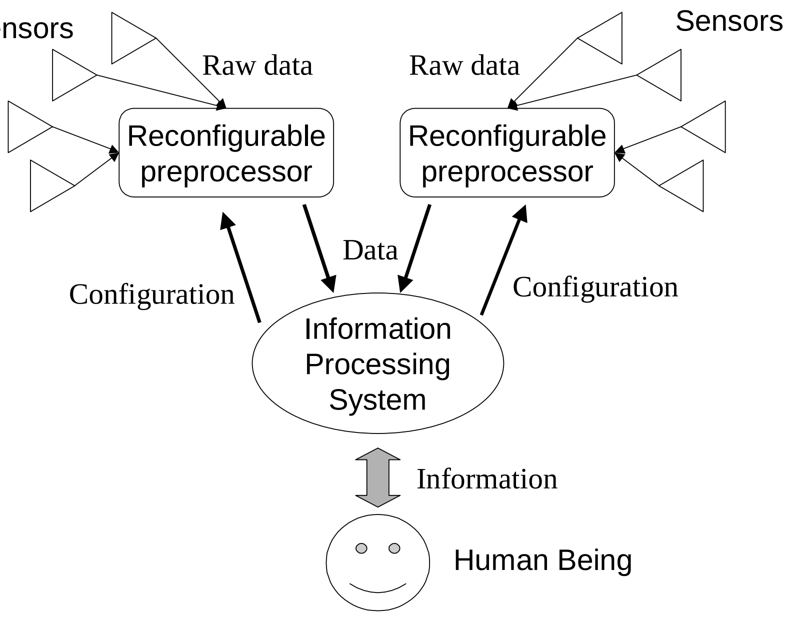
Sensors
Raw data
Raw data
Sensors
Reconfigurable preprocessor
Reconfigurable preprocessor
Data
Configuration
Configuration
Information Processing System
Information
Human Being

Fig.1 Reconfigurable Logic for Intelligent Human Sensing

2. Research Background

This section describes the author's research background and its relation to this project.

2.1 Custom Computing Circuit

The author has been interested in various kinds of custom computing circuits, particularly in those targeted for hard computation problems [1]. The following is a brief summary of custom circuits designed by the author and his colleagues for subgraph isomorphism problems.

Although subgraph isomorphism problems appear to be simple graph problems, they are actually NP-complete and difficult to solve [2]. The most popular algorithm for this problem is Ullmann's algorithm [3], which includes an effective pruning procedure (*refinement procedure*). Ullmann pointed out that his procedure could be implemented by parallel hardware. However, the author's group revealed that his design is too large for actual implementation [4]. Other researchers proposed a new algorithm for subgraph isomorphism (Konishi's algorithm), which is simpler and more cost-effective than Ullmann's [4]. This algorithm was actually implemented on an FPGA, which outperformed the software of Ullmann's algorithm on an off-the-shelf personal computer [5][6].

2.2 Dynamic Reconfiguration

FPGA is usually configured in the system initialization, and that configuration is used while the system is operational. However, some FPGA products can be reconfigured at run-time to adapt to the applications. This kind of run-time reconfiguration is called "dynamic reconfiguration."

The author once built a dynamically reconfigurable logic board, OPERL [7]. OPERL is a PCI card that can be mounted on any personal computer or workstation that offers PCI bus slots. An OPERL board includes two chips of Lucent ORCA 2C FPGA that maximally contain 40K logic gates. The user can reconfigure the logic in 10 ms, which is short enough compared to the execution time of the applications.

Dynamic reconfiguration is a technique for utilizing the logic resources efficiently. In ordinary systems, each logic element has its function fixed. Since all elements do not operate simultaneously, the hardware resources are only partially in use at any moment. By contrast, in dynamic reconfigurable systems, the logic resources are reusable and configurable in response to the resource requirements at each moment. For example, assume that many multiplications are required at one stage of a program. With reconfigurable hardware, we can configure as many multipliers as possible in hardware, thus accelerating the execution of the program. Subsequently, these resources are reclaimed and reconfigured for other purposes.

Dynamic reconfiguration can also contribute to low power consumption. A system usually includes many logic components with various functions, but they are not used simultaneously. Many units just remain unused, needlessly consuming power. If dynamic reconfiguration is adopted, we can avoid futile power consumption by implementing only those units that are really necessary at that moment. Far from wishful thinking, this system has already been adopted in a low-power consumer product from Sony [8].

2.3 Data Dependent Circuit

In the previous paragraphs, we discussed customizing logic circuits for an application or an algorithm. Here, it should be noted that an application program consists of an algorithm and input data. If we can reconfigure hardware for an algorithm, why can't we configure it also for input data? It is actually possible to customize hardware for data, just as a traditional computer (von Neumann computer) stores both the instruction sequence and the input data in the same main memory.

This is not an issue specific to reconfigurable hardware. Even in conventional software, we can generate an input-specific program from an original program and its input data. This technique is called *software specialization* or *partial evaluation* [9], which makes a program smaller and faster.

In the logic design, we can generate *data dependent circuits* to derive smaller and faster circuits. The basis of this idea is as follows: If any input of a logic gate is fixed to a constant, that gate is eliminated, and the corresponding constant propagates to the output of that gate. Such reduction is recursively applicable, consequently reducing many combinatorial gates and flip-flops. The derived circuit will operate at a higher frequency because the length of the critical path is also reduced.

The major drawback of this technique is that the circuit becomes partially or wholly input specific, i.e., not reusable. Hence, we have to reckon the circuit generation time in addition to the execution time itself. It depends on each case whether or not the data dependent circuit proves advantageous. The authors examined data dependent circuits for subgraph isomorphism problems in a quantitative manner [10][11]. According to our results, they are effective in case of larger problems where the execution time increases very quickly.

2.4 Load Balancing for Heterogeneous Systems

In heterogeneous systems, load balancing is very important for optimal performance. Even if the sub-units are optimally designed, the total performance can be spoiled by bottlenecks between sub-units. In other words, each sub-unit should be designed considering the overall balance. Otherwise, the resulting system would be unbalanced and inefficient for its cost.

The authors have studied various aspects of parallel and distributed systems, including load-balancing schemes for scientific applications [12][13][14][15]. In our scheme, the execution time is modeled as a mathematical programming problem, taking both the communication time and the calculation time into consideration. This problem is treated as a combinatorial optimization problem to be solved so as to make the total execution time optimal. We also dealt with a distributed computing environment, in which the processing elements are not necessarily uniform. Such heterogeneous computing clusters pose difficult problems for load balancing because they involve a large number of free variables. Nevertheless, we showed that we can solve the problem for a realistic number of processors. We hope that our approach is applicable to heterogeneous multiprocessors for this intelligent human sensing project.

3. Future Studies

The author plans to explore circuit generation techniques suitable for sensor preprocessors. The self-organization of custom-logic circuits would also be an interesting and ambitious research subject. Such a study inevitably entails evaluations on actual FPGA platforms. We have already prepared a couple of evaluation boards for various purposes, and would use them in subsequent studies. We hope to tackle both laboratory research problems and real-world industrial applications.

Acknowledgments

Our studies have been partially supported by a Grant-in-Aid for Scientific Research from the Japan Society for the Promotion of Science (JSPS), as well as by grants from the Telecommunications Advancement Foundation (TAF), the Hori Information Science Promotion Foundation, and the Okawa Foundation for Information and Telecommunications. This work is also supported in the 21st Century COE Program ``Intelligent Human Sensing" from the Ministry of Education, Culture, Sports, Science and Technology. The custom circuits in our studies were designed with Synopsys CAD tools through the chip fabrication program of VLSI Design and Education Center (VDEC), the University of Tokyo.

References

[1] S. Ichikawa: "Custom Computing Machinery for Hard Computation Problems," Algorithm Engineering (K. Sugihara, et al. Ed.), Kyoritsu Shuppan, Section 6.27, pp. 270—271 (2001). (in Japanese)

[2] M.R. Garey, D.S. Johnson: "Computers and Intractability," Freeman (1979).

[3] J.R. Ullmann: "An Algorithm for Subgraph Isomorphism," J. ACM, Vol.23, No.1, pp.31—42, 1976.

[4] S. Ichikawa, H. Saito, L. Udorn, K. Konishi: "Evaluation of Accelerator Designs for Subgraph Isomorphism Problem," Proc. 10th Int'l Conf. Field Programmable Logic and Applications (FPL 2000), LNCS 1896, Springer, pp. 729—738 (2000).

[5] S. Ichikawa, L. Udorn, K. Konishi: "An FPGA-Based Implementation of Subgraph Isomorphism Algorithm," IPSJ Transactions on High Performance Computing Systems, Vol. 41, No. SIG5 (HPS1), pp. 39—49 (2000). (in Japanese)

[6] S. Ichikawa, L. Udorn, and K. Konishi: "Hardware Accelerator for Subgraph Isomorphism Problems," Proc. Eighth IEEE Symposium Field-Programmable Custom Computing Machines (FCCM'00), pp. 283—284 (2000).

[7] S. Ichikawa and T. Shimada: "Reconfigurable PCI Card for Personal Computing," Proc. 5th FPGA/PLD Design Conference & Exhibit, pp. 269—277 (1997). (in Japanese)

[8] Sony Corp.: "Sony develops "Virtual Mobile EngineTM", A new reconfigurable circuit technology for much lower power-consumption Adopted in Network Walkman's LSI," http://www.sony.net/SonyInfo/News/Press/200212/02-1210E/.

[9] C. Consel, O. Danvy: "Tutorial Notes on Partial Evaluation", Proc. 20th ACM Symp. Principles of Programming Language, pp. 493—501 (1993).

[10] S. Ichikawa, S. Yamamoto: "Data Dependent Circuit for Subgraph Isomorphism Problem," IEICE Transactions on Information and Systems, Vol. E86D, No. 5 (2003). (to appear).

[11] S. Ichikawa, S. Yamamoto: "Data Dependent Circuit for Subgraph Isomorphism Problem," Proceedings of 12th Int'l Conf. on Field Programmable Logic and Applications (FPL 2002), LNCS 2438, Springer, pp. 1068—1071 (2002).

[12] S. Ichikawa, S. Yamashita: "Static Load Balancing of Parallel PDE Solver for Distributed Computing Environment," Proc. ISCA 13th Int'l Conf. Parallel and Distributed Computing Systems (PDCS-2000), pp. 399—405 (2000).

[13] S. Ichikawa, S. Yamashita: "Static Load-balancing for Distributed Processing of Numerical Simulations," IPSJ Journal, Vol. 42, No. 3, pp. 552—564 (2001). (in Japanese)

[14] S. Ichikawa, Y. Fujimura: "Iterative Data Partitioning Scheme of Parallel PDE Solver for Heterogeneous Computing Cluster," Proc. IASTED Int'l Conf. Applied Informatics: Int'l Symp. Parallel and Distributed Computing and Networks, ACTA Press, pp. 364—369 (2002).

[15] Y. Kishimoto, S. Ichikawa: "Parallel Linpack Benchmark Results on Heterogeneous Cluster," Proceedings of JSPP2002, pp. 177—178 (2002). (in Japanese)