Mono-instruction computer on a dynamically reconfigurable gate array

Yuki Nihira and Minoru Watanabe
Electrical and Electronic Engineering
Shizuoka University
3-5-1 Johoku, Hamamatsu, Shizuoka 432-8561, Japan
Email: tmwatan@ipc.shizuoka.ac.jp

Abstract— As gates in field programmable gate arrays (FPGAs) become usable in ever-increasing numbers, FPGAs are becoming more widely used in various applications. Currently, FPGAs are implemented in many embedded systems. Demand for implementing a processor onto an FPGA is gaining. In response to that demand, FPGA vendors have provided soft-core processors for FPGAs, but those processors invariably have lower performance than that of hard-core processors. This paper therefore presents a proposal for a high-performance mono-instruction computer that fully exploits the programmability of a dynamically reconfigurable gate array. In addition, this paper clarifies implementation area and operation frequency advantages of mono-instruction computers relative to soft-core RISC processors.

I. Introduction

Field programmable gate arrays (FPGAs) offer increasingly numerous gates for use in widely various applications [1][2]. Currently, FPGAs are implemented onto many embedded systems, and demand for implementing a processor onto an FPGA is gaining. To satisfy that demand, FPGA vendors have come to provide soft-core processors for FPGAs [3]–[6]. Altera Corp. provides a NIOS processor [3][4], while XILINX Inc. provides its Micro Blaze processor [5][6]. However, the soft-core processors invariably have lower performance than hard-core processors inside FPGAs, Intel's processors, ARM processors [7][8] and so on. Therefore, a hard-core processor or an external processor chip must invariably be implemented in addition to an FPGA to produce high-performance embedded systems.

Such low performance of soft-core processors on FPGAs results from its look-up table (LUT) and switching matrix (SM) architecture. In a custom processor, dedicated logic circuits and metal wires are used, respectively, instead of such LUTs and SMs. Therefore, it is difficult for an FPGA's soft-core processor under the same static implementation to surpass the performance of custom processors. However, if the programmability of FPGAs can be exploited, then the performance of its programmable gate array can be increased. The idea is based on high-speed dynamic reconfiguration. The concept was presented in the paper [9]. However, unfortunately, current FPGAs cannot support such high-speed dynamic reconfiguration.

Therefore, to realize such dynamic high-speed reconfiguration, optically reconfigurable gate arrays (ORGAs) have been developed [10]–[19] as shown in Fig. 1. An ORGA consists of a holographic memory, a laser array, and an optically reconfigurable gate array VLSI. Circuit information or configuration contexts can be stored on a holographic memory and can be addressed using a laser array. Finally, they can be optically programmed onto an optically reconfigurable gate array. The optically reconfigurable gate array architecture realizes clock-by-clock high-speed nanosecond-order reconfiguration since the bandwidth of an optical bus between a holographic memory and a programmable gate array VLSI is extremely large. In addition, numerous reconfiguration contexts can be realized since the storage capacity of a three-dimensional holographic memory is larger than those of silicon memories. An ORGA has achieved 100 reconfiguration contexts [14]. A clock-by-clock high-speed reconfiguration infrastructure is mature by the ORGA architectures.

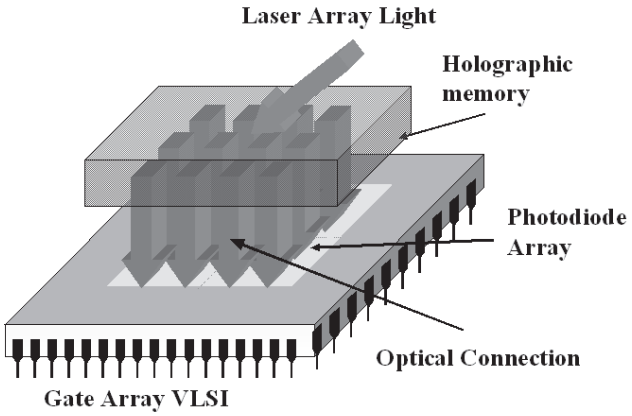

Fig. 1. Overview of an ORGA.

This paper therefore presents a demonstration result of a mono-instruction computer that completely exploits the programmability of a dynamically reconfigurable gate array. Furthermore, this paper clarifies the total performance improvement of mono-instruction computers relative to that of a soft-core RISC processor.

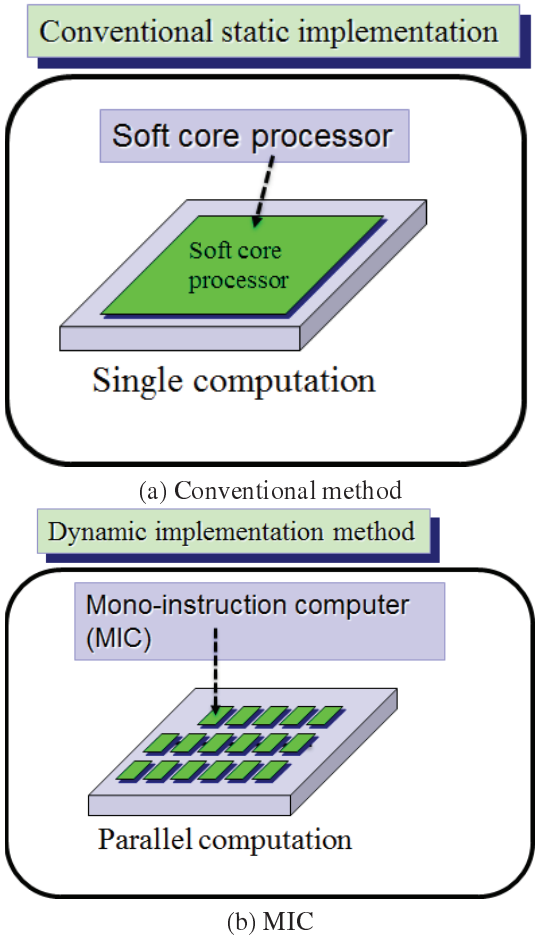

Fig. 2. Concept of a mono-instruction computer (MIC).

II. Mono-instruction computer (MIC)

Recently, almost all computer systems use reduced instruction set computer (RISC) architectures [20][21]. Such architectures offer benefits in terms of higher clock frequency, smaller implementation area, and lower power consumption than conventional complex instruction set computer (CISC) architectures [22][23]. Their success is based on the fundamental principle that the simplest circuit can function with the highest clock frequency, in the smallest implementation area, and with the lowest power consumption because the simplest circuit can be constructed with fewer selector passes, less load capacitance of fewer gates, and less capacitance of short metal wires. This principle is also applicable to fine-grained programmable gate arrays such as FPGAs. The ultimately simplest processor type is a mono-instruction computer (MIC) [9][24]. Only those necessary for a single clock cycle can be implemented onto the programmable device if clock-by-clock reconfiguration for a programmable device is possible. Such a processor can operate at the highest clock frequency, with the lowest power consumption, and in the smallest implementation area. Moreover, the extremely small implementation area enables large parallel computation when using equal implementation areas to those provided by conventional processors, as shown in Fig. 2. Consequently, the overall performance can be increased dramatically.

Therefore, MIC represents a part of a conventional RISC processor. A MIC includes only a single instruction. When realizing various functions exactly like those of a conventional RISC, various MICs are prepared. For example, one MIC has an adder instruction, one MIC has a subtractor instruction, one MIC has a multiplier instruction, one MIC has a divider instruction, and so on. Every MIC's operation is completed in a single clock cycle.

Therefore, for example, when a sequence operation of an adder operation, a subtractor operation, and a multiplier operation is required, an adder MIC is first configured onto a programmable device. Then the operation is executed at the first clock cycle and the calculation result is left on a register. Here, during reconfiguration, the register's information is maintained as a previous state. Then, during the subsequent clock cycle, a subtractor MIC is configured onto the adder MIC implementation area of the programmable device, then the subtractor operation is executed based on the register's retained information. Finally, the result is stored on the register. Then, at the third clock cycle, a multiplier MIC is configured onto the subtractor MIC implementation area of the programmable device, and the multiplier operation is executed based on the left register's information. Finally, the result is stored on the register.

As described above, under the MIC implementation, a necessary MIC is implemented dynamically at a necessary time. The instruction change is based on the MIC dynamic reconfiguration. Each MIC requires small implementation area, so a large parallel computation is also expected. In this case, the total performance is increased dramatically. To date, only the implementation area reduction has been clarified [24]. In addition to the implementation area reduction, total performance analysis including the clock frequency advantage and parallel computation advantage based on the area reduction has never been reported in the literature.

Therefore, this study was undertaken to estimate the total performance improvement of mono-instruction computers in terms of clock frequency, implementation area, and parallel computation, compared with a soft-core RISC processor. High-performance MIC implementation merely requires a high-speed clock-by-clock reconfigurable device, exactly like optically reconfigurable gate arrays (ORGAs). Currently, the hardware infrastructure based on ORGAs is mature.

III. Experiment environment

ORGA logic synthesis tools and place and route tools are in a development phase [25]. Therefore, to estimate the mono-instruction computer (MIC), we used a DE2-70 Board (Altera Corp.) for this study. The board's FPGA (Cyclone II 2C70 FPGA; Altera Corp.) includes 68,416 logic elements, 250 M4K RAM blocks, 150 embedded multipliers, 4 PLLs, and 622 user I/O pins. However, in this experiment, no embedded multipliers, M4K RAM blocks, or PLLs were used. Only

logic elements were used for implementing the MICs. The implemented results of 11 MICs are presented in Table 1. A frequency report for each MIC is provided by software (Quartus II Web Edition, ver. 11.0; Altera Corp.). In this estimation, an arithmetic and logic unit (ALU) of a 32 bit original soft-core RISC processor was first designed for comparison with the MICs. The 32-bit soft-core RISC processor's ALU includes all MIC functions: a 32-bit adder function, a 32-bit multiplier, a 32-bit subtractor, a 32-bit divider, 32-bit logical functions, and 32-bit barrel shifters. The ALU was implemented on the FPGA (Cyclone II 2C70; Altera Corp.), as shown in the last line of Table 1, as well as MICs. The logic synthesis, place, and route were also executed using the Quartus II Web Edition Software. The ALU's maximum clock frequency was reported as 8.11 MHz. The requirement logic cells were 2,523. This conventional soft-core processor implementation example for an FPGA with no reconfiguration is used for comparison of the MIC performance with that of the conventional soft-core implementation method.

IV. MIC IMPLEMENTATIONS

Each MIC was implemented onto the same implementation area, 2,523 logic cells on the Cyclone II 2C70 FPGA, for comparison with the soft-core processor described above.

A. 32-bit adder MIC

A 32-bit adder MIC has only a single 32-bit adder function. As the first line of Table 1 shows, a single 32-bit adder MIC was implemented on 99 logic cells. At that time, the maximum clock frequency was 64.65 MHz. For the soft-core RISC processor described above, the implementation area reached 2,523 logic cells. Therefore, using the same number of logic cells as the soft-core RISC processor, 25.5 MICs can be implemented onto the FPGA and can be executed in parallel so that 25.5 times better performance can be achieved. Moreover, the clock frequency of each 32-bit adder MIC is 7.97 times higher than that of the soft-core RISC processor. Consequently, the total performance can be estimated as 203.2 times better than that of a conventional soft-core RISC processor.

B. 32-bit multiplier MIC

A 32-bit multiplier MIC was estimated as shown in the second line of Table 1. The 32-bit multiplier MIC has only a single 32-bit multiplier function with two-32 bit inputs and a 32-bit output. The MIC has consumed 527 logic cells on the FPGA. At that time, the timing report of the Quartus II tool showed that the maximum clock frequency was 46.93 MHz. With the same number of logic cells as the soft-core RISC processor, 4.8 multiplier MICs can be implemented onto the FPGA and can be executed in parallel so that 4.8 times better performance can be achieved. The clock frequency is 5.79 times higher than that of the soft-core RISC processor. Finally, the total performance can be estimated as 27.7 times better than that of conventional the soft-core RISC processor.

C. 32-bit subtractor MIC

As well, a 32-bit subtractor MIC was estimated as shown in the third line of Table 1. The 32-bit subtractor MIC has consumed 99 logic cells on the FPGA. The maximum clock frequency was reported as 66.05 MHz. Under the same number of logic cells as the soft-core RISC processor, 25.5 32-bit subtractor MICs can be implemented onto the FPGA and can be executed in parallel so that 25.5 times better performance can be achieved. The clock frequency is 8.14 times higher than that of the RISC. Finally, the total performance can be estimated as 207.6 times better than that of conventional the soft-core RISC processor.

D. 32-bit divider MIC

A 32-bit divider MIC was estimated as shown in the fourth line of Table 1. The 32-bit divider MIC used 1,145 logic cells. The maximum clock frequency was 8.19 MHz. Under the same number of logic cells as those of the soft-core RISC processor, 2.2 divider MICs can be implemented onto the FPGA and can be executed in parallel so that 2.2 times higher performance can be achieved. The clock frequency is about 1.01 times higher than that of the RISC. Finally, the total performance is estimated as 2.2 times better than that of the conventional soft-core RISC processor.

E. 32-bit logical function MIC

Furthermore, 32-bit logical function MICs were estimated as shown in the fifth to the eighth lines of Table 1. The AND MIC required 64 logic cells. The maximum clock frequency is 420.17 MHz. Under the same conditions as those for the soft-core RISC processor, 39.4 AND MICs can be implemented onto the FPGA and can be executed in parallel, thereby achieving 39.4 times higher performance. The clock frequency is about 51.8 times higher than that of the RISC. Finally, total performance can be estimated as 2,042.4 times better than that of the conventional soft-core RISC processor. The performance of the other functions are same as this result.

F. 32-bit barrel shifter MIC

In addition, 32-bit barrel shifter MICs were estimated as shown in the ninth to the eleventh lines of Table 1. The 32-bit barrel shifter MICs used 245–248 logic cells. The maximum clock frequencies were 181.69–216.08 MHz. Under the same conditions as those of the soft-core RISC processor, 10.2–10.3 barrel shifter MICs can be implemented onto the FPGA and can be executed in parallel so that 10.2–10.3 times higher performance can be achieved. The clock frequency is about 22.4–26.6 times faster than that of the RISC. Finally, its total performance can be estimated as 229.8–274.3 times better than that of a conventional soft-core RISC processor.

TABLE I
MONO-INSTRUCTION COMPUTER (MIC) IMPLEMENTATION AND A CONVENTIONAL SOFT-CORE PROCESSOR IMPLEMENTATION ON THE FPGA

Implemented Circuit	Logic Cells	Clock Frequency [MHz]	Total performance ratio (MIC/RISC)
32 bit Adder MIC	99	64.65	203.2
32 bit Subtracter MIC	99	66.05	207.6
32 bit Multiplier MIC	527	46.93	27.7
32 bit Divider MIC	1,145	8.19	2.2
32 bit AND MIC	64	420.17	2,042.4
32 bit OR MIC	64	420.17	2,042.4
32 bit EXOR MIC	64	420.17	2,042.4
32 bit Inverter MIC	64	420.17	2,042.4
Barrel Shifter MIC (left,zero extension)	248	200.52	251.5
Barrel Shifter MIC (right,sign extension)	245	216.08	274.3
Barrel Shifter MIC (right,zero extension)	246	181.69	229.8
32 bit soft-core processor	2,523	8.11	1

V. DISCUSSION AND CONCLUSION

As usable gates on an FPGA become increasingly numerous, FPGA uses for various applications are undergoing rapid and broad expansion. Currently, FPGA vendors provide soft-core processors for FPGAs. However, the soft-core processor performance is lower than that of a hard-core processor inside an FPGA, an ARM processor, an Intel processor, and so on. This paper therefore has presented estimation results of a mono-instruction computer (MIC) implementation that fully exploits the programmability of a dynamically reconfigurable gate array. Results show that, when using the Cyclone II 2C70 FPGA, the MIC performance was estimated as 2.2-2,042.4 times better than that of a conventional soft-core RISC processor's static implementation. Of course, since the Cyclone II 2C70 FPGA cannot be dynamically reconfigured, numerous long idle times occur between the MIC executions. For that reason, the MIC is not a practical implementation. However, currently, high-speed dynamically reconfigurable devices, ORGAs have been developed to support a non-overhead clock-by-clock nanosecond reconfiguration. The infrastructure for MICs is being put into place.

ACKNOWLEDGMENTS

This research was supported by the Ministry of Education, Science, Sports and Culture, Grant-in-Aid for challenging Exploratory Research, No. 23650087. The VLSI chip in this study was fabricated in the chip fabrication program of VLSI Design and Education Center (VDEC), the University of Tokyo in collaboration with Rohm Co. Ltd. and Toppan Printing Co. Ltd.

REFERENCES

[1] Altera Corporation, "Altera Devices," http://www. altera.com.

[2] Xilinx Inc., "Xilinx Product Data Sheets," http://www. xilinx.com.

[3] J. Perez Acle, M. S. Reorda, M. Violante, "Implementing a safe embedded computing system in SRAM-based FPGAs using IP cores: A case study based on the Altera NIOS-II soft processor," IEEE Second Latin American Symposium on Circuits and Systems, pp. 1-5, 2011.

[4] O. A. Al Rayahi, M. A. S. Khalid, "UWindsor Nios II: A soft-core processor for design space exploration," IEEE International Conference on Electro/Information Technology, pp. 451 - 457, 2009.

[5] R. H. Klenke, "Experiences Using the Xilinx Microblaze Softcore Processor and uCLinux in Computer Engineering Capstone Senior Design Projects," IEEE International Conference on Microelectronic Systems Education, pp. 123 - 124, 2007.

[6] A. Klimm, O. Sander, J. Becker, "A MicroBlaze specific co-processor for real-time hyperelliptic curve cryptography on Xilinx FPGAs," IEEE International Symposium on Parallel & Distributed Processing, pp. 1 - 8, 2009.

[7] G. R. Allen, G. M. Swift, G. Miller, "Upset Characterization and Test Methodology of the PowerPC405 Hard-Core Processor Embedded in Xilinx Field Programmable Gate Arrays," IEEE Radiation Effects Data Workshop, Vol. 0, pp. 167 - 171, 2007.

[8] R. Ali, R. Radhakrishnan, G. Kochhar, J. Hsieh, O. Celebioglu, K. Chadalavada, R. Rajagopalan, "Evaluating performance of BLAST on Intel Xeon and Itanium2 processors," IEEE International Workshop on Workload Characterization, pp. 81-88, 2004.

[9] M. Watanabe, F. Kobayashi, "Optically Reconfigurable Gate Arrays vs. ASICs," IEEE Asia Pacific Conference on Circuits and Systems, pp. 1166-1169, 2006.

[10] J. Mumbru, G. Panotopoulos, D. Psaltis, X. An, F. Mok, S. Ay, S. Barna, E. Fossum, "Optically Programmable Gate Array," SPIE of Optics in Computing 2000, Vol. 4089, pp. 763–771, 2000.

[11] J. Mumbru, G. Zhou, X. An, W. Liu, G. Panotopoulos, F. Mok, and D. Psaltis, "Optical memory for computing and information processing," SPIE on Algorithms, Devices, and Systems for Optical Information Processing III, Vol. 3804, pp. 14–24, 1999.

[12] J. Mumbru, G. Zhou, S. Ay, X. An, G. Panotopoulos, F. Mok, and D. Psaltis, "Optically Reconfigurable Processors," SPIE Critical Review 1999 Euro-American Workshop on Optoelectronic Information Processing, Vol. 74, pp. 265-288, 1999.

[13] M. Nakajima, M. Watanabe, "A four-context optically differential reconfigurable gate array," IEEE/OSA Journal of Lightwave Technology, Vol. 27, No 20, pp. 4460-4470, 2009.

[14] M. Nakajima, M. Watanabe, "A 100-Context Optically Reconfigurable Gate Array," IEEE International Symposium on Circuits and Systems, pp. 2884-2887, 2010.

[15] M. Nakajima, M. Watanabe, "36-context dynamic optically reconfigurable gate array," IEEE International Symposium on System Integration, pp. 19-23, Tokyo, Japan, Dec., 2009.

[16] M. Watanabe, F. Kobayashi, "Dynamic Optically Reconfigurable Gate Array," Japanese Journal of Applied Physics, Vol. 45, No. 4B, pp. 3510-3515, 2006.

[17] M. Miyano, M. Watanabe, F. Kobayashi, "Optically Differential Reconfigurable Gate Array," Electronics and Computers in Japan, Part II, Issue 11, vol. 90, pp. 132-139, 2007.

[18] M. Watanabe, T. Shiki, F. Kobayashi, "Scaling prospect of optically differential reconfigurable gate array VLSIs," Analog Integrated Circuits and Signal Processing, Vol. 60, Issue 1-2, pp. 137-143, 2009.

[19] D. Seto, M. Watanabe, "A dynamic optically reconfigurable gate array - perfect emulation," IEEE Journal of Quantum Electronics, Vol. 44, Issue 5, pp. 493-500, 2008.

[20] Y. Pizhou, L. Chaodong, "A RISC CPU IP core," International Conference on Anti-counterfeiting, Security and Identification, pp. 356 - 359, 2008.

[21] J. Goodacre, A. N. Sloss, "Parallelism and the ARM instruction set architecture," Computer, Vol. 38, Issue 7, pp. 42 - 50, 2005.

[22] T. Jamil, "RISC versus CISC," IEEE Potentials, Vol. 14, Issue 3, pp. 13-16, 1995.

[23] D. B. Tolley, "Analysis of CISC versus RISC microprocessors for FDDI network interfaces," Conference on Local Computer Networks, pp. 485 - 493, 1991.

[24] F. Kobayashi, Y. Morikawa, M. Watanabe, "MISC: Mono Instruction-Set Computer based on Dynamic Reconfiguration - a 6502 Perspective-," International Conference on engineering of reconfigurable systems and algorithms, pp. 222-228, 2008.

[25] M. Watanabe, F. Kobayashi, "A logic synthesis and place and route environment for ORGAs," International conference on engineering of reconfigurable systems and algorithms, pp. 237-238, 2006.