Configurable Multi-Processor Architecture and its Processor Element Design

Tsutomu Nishimura, Takuji Miki, Hiroaki Sugiura, Yuki Matsumoto, Masatsugu Kobayashi,
Toshiyuki Kato, Tsutomu Eda and Hironori Yamauchi
VLSI center
Ritsumeikan University
1-1-1 Nojihigashi, Kusatsu, Shiga 525-8577, Japan
Email: ro010014@se.ritsumei.ac.jp

Abstract— We developed an application specific multi-processor generation system intended for real-time applications. In this system, we adopted a distributed memory type multi-processor architecture with hierarchical tree network as a configurable multi-processor which can be adapted to various scale systems flexibly. We have also developed a configurable multi-processor prototype as LSI chips with the 0.18 μm CMOS standard cell technology.

I. Introduction

Parallel processing by a multi-processor has been suggested as a solution to the processor performance limit of a single processor [1]. In the traditional multi-processor system, the application program has been scheduled on the fixed hardware. Therefore, the multi-processor's performance has been limited because of structural mismatch between hardware and software.

To solve this problem, we propose the automatic generation system for a multi-processor based on Hardware-Software Co-design [2] [3]. The automatic generation system generates three objectives based on *C* program and HDL source of PE. These three objectives are PEs, network switches and the objective codes of each PE (see Fig.1).

We have studied the multi-processor hardware ardhitecture for the configurable multi-processor that can be adapted for various scale systems. Based on this study we designed and fablicated "it" as an LSI chip.

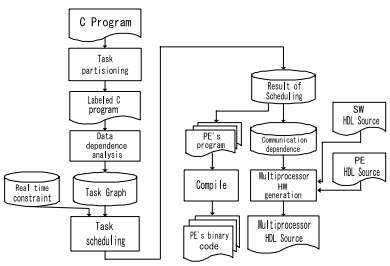

Fig. 1. Automatic generation flow

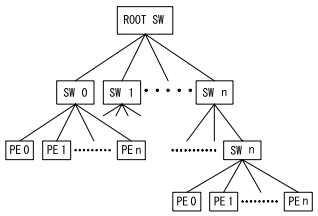

Fig. 2. Multi-processor System

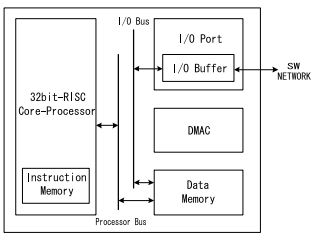

Fig. 3. PE architecture

II. Multi-processor Architecture

We designed the multi-processor architecture to be used for the automatic generation. This system has to be flexible enough to adapt itself to various scale multi-processor systems flexibly. Consequently, we adopted the distributed memory type multi-processor that can be a larger scale multi-processor than the shared memory type.

Figure 2 shows our proposed multi-processor system. In this system, the PE network is a hierarchical tree structure composed as a Matrix Switch. Therefore, this system has high scalability and it is able to form the best structure according to *C* program.

A. PE Architecture

One PE is composed of a 32-bit-RISC Core-processor, Direct Memory Access Controller (DMAC), I/O port,

0-7803-9451-8/06/$20.00

TABLE I
COMPARISON OF PROCESSING PERFORMANCE BETWEEN SINGLE-PROCESSOR AND 16-PE MULTI-PROCESSOR

	Single-processor (the number of clocks)	16 PEs Multi-processor (the number of clocks)	Accelerating Ratio (Multi/Single)
1024-FFT	2,283,931	490,950	4.65 times
DFT	1,931,629	138,648	13.93 times

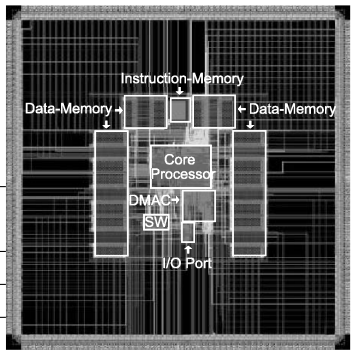

Fig. 4. Layout of the chip

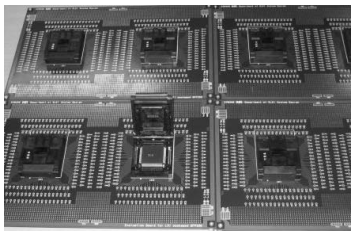

Fig. 5. On board multi-processor system for evaluation

Data-Memory and Instruction-Memory, as shown in Figure 3. The core-processor is based on DLX architecture. In addition to the fundamental instruction set of the RISC architecture, we designed additional instructions that achieve communication processing between PEs. The DMAC and the I/O port are designed to perform communication processing and reduce communication overhead.

III. LSI Implementation

We designed our proposed multi-processor using *Verilog-HDL*, and simulated it with *NC-Verilog* to evaluate its processing performance. In this experiment, we compared the processing performance of the 16-PE multi-processor to that of the single-processor by processing 1024-point FFT program (see table I). We confirmed that the proposed multi-processor processed the FFT and DFT program faster than single processor, up to 14 times as fast as single-processor.

We developed the LSI which is equipped with one PE and one SW as a prytotype for verification and evaluation of the chip. The LSI was fabricated with the 0.18 μm process. The memory is loaded as SRAM (IMEM: 32-bit $\times$ 512 words, DMEM: 32-bit $\times$ 4K words). About 150K transisters are integrated in each PE.

Figure 4 shows the layout of the chip. The die size is 6166 μm $\times$ 6166 μm, and the core size is 3166 μm $\times$ 3166 μm. The chip has 208 pins, which include the power supply pins (VDD: 1.8V and 3.3V, VSS). We back-annotated a functional verification with the timing information of the place and route. This chip can operate up to 50 MHz.

The test circuit is equipped in this LSI. We adopted the direct access method as a typical ad-hoc technique. Using multiplexers, drawing out signals, that can not be accessed directly from outside of the chip, we were able to test the each function block. Therefore, this chip can be used as either PE or SW, and we constructed a multi-processor system using many of them on the evaluation boards (see Figure 5). We achieved a multi-processor system with the capability to process the FFT program, and we verified that these chips can operate at 33 MHz.

IV. Conclusions

In this paper, we have proposed a multi-processor generation system, and designed a configurable multi-processor for this system. The proposed multi-processor system is so flexible that it is able to form the approproate structure for processing various application programs. We confirmed that the proposed multi-processor achieved high parallel processing performance. Following that, we fabricated "it" by using the 0.18 μm CMOS technology and realized a multi-processor system composed of a sufficient number of LSI chips on evaluation boards.

In future work, we will improve our automatic generation system so that it will be applicable to any large scale application programs.

Acknowledgements

The authors would like to thank Mr. Daisuke Okawa from Rohm Corporation for his support.

The LSI chip in this research has been fabricated in Rohm Corporation.

References

[1] W. H. Wolf, "An architectural co-synthesis algorithm for distributed embedded computer systems," *IEEE Trans.*, Veri Large Scale Integration Systems, Vol. 5, No. 2, pp. 218-229, June 1997.

[2] H. El-Rewini, T. G. Lesis and H. H. Ali, "Task scheduling in parallel and distributed systems," pp. 56-81, Prentice Hall, 1994.

[3] H. Nishikado, A. Nishi and H. Yamauchi, "Fast scheduling method for multi-processor using genetic algorithms," Proc. *IEEE ISPACS'99*, pp. 171-174, Dec. 1999.