# Reconfigurable Synchronized Dataflow Processor

Hiroshi Sasaki
Hitoshi Maruyama
Hiroaki Kobayashi
Tadao Nakamura

Graduate School of Information Sciences,
Tohoku University
Sendai, MIYAGI 980-8579
Tel : +81-22-217-7013
Fax : +81-22-216-8113
e-mail : {sasa, maru, koba, nakamura}@archi.is.tohoku.ac.jp

Hideaki Tsukioka
Nobuyoshi Shoji

Friendly Systems Co., Ltd.
AER 810, Chuo 1-3-1,
Sendai, MIYAGI 980-6108
Tel : +81-22-261-3693
Fax : +81-22-713-8170
e-mail : {tsuki, shuho}@friendly-systems.co.jp

**Abstract - This paper describes the design and implementation of a reconfigurable synchronized dataflow processor (RSDP). The RSDP can configure its hardware to directly represent dataflow graphs (DFGs) of applications. Data are processed while they flow along application-specific datapaths in the RSDP. We have designed three DFGs for benchmarking and evaluated their performance on an RSDP board. The results show that the RSDP running at relatively lower frequency can achieve a competitive performance with a general-purpose processor.**

## I. Introduction

Dataflow computation models have been investigated in various points of view [1, 2, 3, 4]. They differ from each other in the number of tokens on an arc, flow control, token identification, and the treatment of function calls. Most previous researches focus on conceptual dataflow models so that their programming model and the functionality of hardware do not match the requirements of applications. Therefore, we propose an RSDP suited for the efficient mapping and execution of applications in DFGs.

The rest of the paper is organized as follows. Section II describes the functional design and the LSI implementation of the RSDP. A software development environment is presented in Section III. We evaluate the performance of the RSDP and compared it with those of a general-purpose processor in Section IV. Section V gives summary and conclusions.

## II. Reconfigurable Synchronized Dataflow Processor

An RSDP consists of functional units (FUs), an interconnect, and an internal memory. Applications are programmed using DFGs where nodes and arcs of a DFG correspond to configurations for FUs and the interconnect, respectively.

TABLE I
FUs and Their Functions

| FU | Function |
|---|---|
| IOU | Input, Output |
| BCU | Communication between FU blocks |
| MRU | Memory read |
| MWU | Memory write |
| ALU | Addition/Subtraction (Result, Borrow, Carry), Accumulation, Comparison, Logical operations |
| MDU | Multiplication, Division |
| SFU | Shift operations |
| FCU | Flow control operations |
| CGU | Constant generation |
| CTU | Count operations |
| DLU | Delay |

Although implementing all of the possible operations on every unit provides flexibility for node allocation to FUs, it will cause a noticeable increase in hardware cost. Therefore, we classify operations according to the similarity of their logic circuits and implement each class as an FU. Table I shows the list of FUs and their functions.

The block diagram of the RSDP is illustrated in Fig. 1. The FUs communicate each other through the interconnect. The configuration register controls the FUs and the interconnect. An entry of the configuration register corresponds to the configuration of an FU. Each entry has at least one of the fields to specify function, source FUs of incoming data, or other kinds of parameters like constant.

Since the interconnect is expected to be as flexible as possible under the constraint of hardware budget, we limit its width to 1-bit. The limitation contributes to a reduction in the amount of required hardware for both the interconnect and FUs at the expense of increase in latency. FUs transfer data bit-serially through the interconnect. The operations of each bit of a datum are pipelined.

The interconnect comprised of a crossbar network provides flexibility for node allocation to FUs. The constraint on the hardware budget restricts the number of

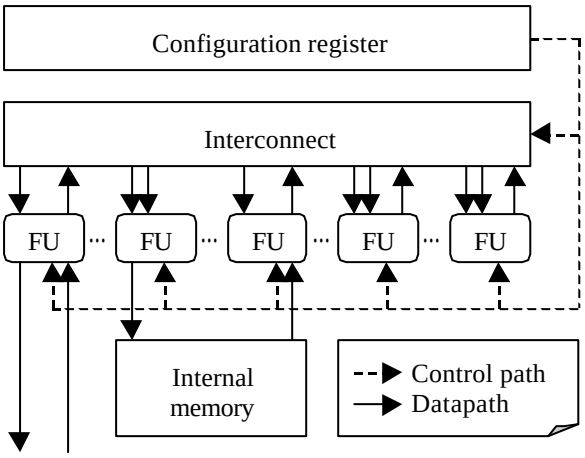


Fig. 1 Block diagram of the RSDP.

nodes in a crossbar network. We employ a hierarchical network of unit blocks in the current implementation to meet the resource constraint. FUs are grouped into 3 blocks, where each block has a crossbar network. For the purpose of data transfer between the FU blocks, we use FUs for inter-block communication (BCU).

A datum consists of a 16-bit literal and a flag. The flag is called valid bit and guarantees the validity of the corresponding literal. The valid bit is transferred first, and then the literal is transferred from LSB to MSB. Transferring a datum, including a valid bit and a 16-bit literal, consumes 17 clock cycles. These 17 clock cycles define an operation cycle of the FUs.

Memory access operations are provided as functions implemented in FUs. An FU for memory read receives an address as input, and then outputs the value in the internal memory specified by the address. On the other hand an FU for memory write receives an address and a value, and then the value is stored in the address of the internal memory.

An LSI chip of the RSDP is fabricated in a 0.35um process technology, consisting of a polysilicon layer and 3 metal layers. Its die size is 9.2mm x 9.2mm with 800K gates. The RSDP chip is designed to operate at 30MHz where the interconnect becomes a critical path. At present, an RSDP board consists of 4 RSDP chips in a cascade, an external memory, a PCI Bridge, and an I/O Controller. The overall RSDP board operates at 13.5 MHz because of the limited performance of the I/O controller.

## III. Software Development Environment

We developed a GUI-based dataflow editor and a simulator as a software development environment called "Soryu." Design sequence of DFGs consists of node placement, function assignment, and node connection. After that, DFGs are simulated and debugged. Finally, a DFG is mapped onto the RSDP, where nodes are allocated to corresponding FUs.

The dataflow editor generates the configurations for the RSDPs directly from a designed DFG and information of node allocation. The task for application development is greatly reduced in the software development environment.

## IV. Performance Evaluation

We have designed DFGs of an RGB-YCbCr color system converter, a color inverse filter, and a 3x3 median filter to evaluate the performance of the RSDP board. The size of the target image is 640x480 pixels, where each pixel consists of 3 bytes (24-bit color).

Table II shows the performance comparison between the RSDP at 13.5 MHz and a AMD K6 at 450 MHz. Frequency-normalized performance ratio (FNPR) shows that the performance of the RSDP is 10 times higher than that of the K6. We believe that the RSDP can outperform general-purpose processors if it runs at higher frequency with the optimization of the RSDP circuit.

TABLE II
Performance Comparisons

| Benchmark \ Processor | RSDP (ms) | AMD K6 (ms) | FNPR (RSDP/K6) |
|---|---|---|---|
| RGB-YCbCr converter | 210 | 90 | 14.0 |
| Color inverse filter | 80 | 40 | 16.6 |
| Median filter | 410 | 190 | 15.0 |

## V. Summary and Conclusions

In this paper, we have described the LSI implementation of the RSDP and the dataflow editor as a software development environment. The performance of a prototype has been measured using three benchmarks. The result shows that the RSDP can achieve over 10 times higher performance than a general-purpose processor if both run at the same frequency.

## Acknowledgements

The LSI implementation was supported by VSAC (Venture System LSI Assist Center) and TSMC (Taiwan Semiconductor Manufacturing Co., Ltd.). This research was financially supported in part with the Miyagi Prefectural Government's subsidy program for creative technology research and development.

## References

[1] J. B. Dennis, "Data Flow Supercomputers," *Computer*, Vol. 13, pp. 48-56, 1982.
[2] J. R. Gurd, "The Manchester Prototype Dataflow Computer," *Communications of the ACM*, vol. 28, No. 1, January 1985.
[3] G. M. Papadpoulos, *Implementations of a General-Purpose Dataflow Multiprocessor*, MIT Press, 1991.
[4] S. S. Battacharyya, R. K. Murthy, and E. A. Lee, *Software Synthesis from Dataflow Graphs*, Kluwer Academic Publishers, 1996.