An Implementation of Pipelined Prallel Processing System for Multi-Access Memory System

Hyung Lee[1], Hyeon-Koo Cho[2], Dae-Sang You[1], and Jong-Won Park[1]
[1] Department of Information & Communications Engineering,
Chungnam National University,
220 Gung-Dong Yusung-Gu, Daejeon, 305-764, KOREA
Tel.: +82-42-821-7793, Fax.: +82-42-825-7792
[2] Virtual I Tech. Inc.,
Room 503, Engineering Building 3,
220 Gung-Dong Yusung-Gu, Daejeon, 305-764, KOREA
e-mail : {hyung, dsyou, hkcho, jwpark}@crow.cnu.ac.kr

Abstract: We had been developing the variety of parallel processing systems in order to improve the processing speed of visual media applications. These systems were using multi-access memory system(MAMS) as a parallel memory system, which provides the capability of the simultaneous accesses of image points in a line-segment with an arbitrary degree, which is required in many low-level image processing operations such as edge or line detection in a particular direction, and so on. But, the performance of these systems did not give a faithful speed because of asynchronous feature between MAMS and processing elements.

To improve the processing speed of these systems, we have been investigated a pipelined parallel processing system using MAMS. Although the system is considered as being the single instruction multiple data(SIMD) type like the early developed systems, the performance of the system yielded about 2.5 times faster speed.

1. Introduction

There already exit the variety of machines that are capable of performing high-speed image applications. In general, these machines can be divided into two classes[1]: the first basically comprises two dimensional(2-D) array processors that operate on an entire image or subimage in a set of parallel processes. Examples of this type of machine are CLIP, MPP, and PIXIE-5000. In general, all of these machines can also be considered as being the single instruction multiple data (SIMD) type. The main drawback of 2D array processors is their cost. In addition, due to the inherent serial nature of the input-image data, full utilization of the processors may not be attained.

The second class of machines – local-windows processors – scans an image and performs operations on a small neighborhood window. Examples of such machines include MITE, PIPE, and Cytocomputer. Note that, with this type of processor, an increase in image size requires a quadratic increase in processor speed in order to maintain a constant processing speed.

Most of the above local-window processors are general purpose in nature in that they are programmable. Although these general-purpose cellular machines are flexible because of their programmability, they do not provide the capability of the simultaneous accesses of image points in a line-segment with an arbitrary degree, which is required in many low-level image processing operations such as edge or line detection in a particular direction, and so on. We have been developing the parallel processing system which is considered as being the pipelined SIMD type with a multi-access memory system(MAMS) satisfying to provide the capability of the simultaneous accesses of image points.

In this paper, we propose 5-stage pipelined parallel processing system involving MAMS for accessing the data elements within three access types with a constant interval simultaneously. Each processing element(PE) is designed with 2 states for which memory access instructions and general instructions. Although two states are performed in parallel, the cycle of general instructions is depended on one of memory access instructions because memory access instructions have to access data via MAMS.

The remainder of this paper is organized as follows. Section 2 introduces multi-access memory system which is redesigned for the proposed system, and the proposed pipelined parallel processing system is described in Section 3. Section 4 presents experimental results yielded by simulations. Finally, we conclude this paper in Section 5 followed by the references.

2. Multi-Access Memory System

For a parallel processing system with PEs, it is necessory to use an MAMS[2,3] to reduce the memory access time. Also, the memory system has the important goals to provide the efficient utilization for $n(= pq)$ PEs of the pipelined parallel processing system we proposed, where p and q are design parameters. The goals are as follows: various access types and constant interval between the data elements, simultaneous access with no restriction on the location, simple and fast address calculation and routing circuitry, and small number of memory modules.

The memory system consists of a memory module selection circuitry, a data routing circuitry for WRITE, an address and a routing circuitry, memory modules, and a data routing circuitry for READ. In order to distribute the data elements of the $M \times N$ array $I(*,*)$ among $m(= pq + 1)$ memory modules, a memory module assignment function must place in distinct memory modules array elements that are to be accessed simultaneously. Also, an address assignment function must allocate different

addresses to array elements assigned to the same memory module.

The MAMS we redesigned is implemented to the pipelined with three stages because the parallel processing system will be introduced in Section 3 is designed for pipelined architecture. In the case of sequential memory operations, therefore, memory access times are reduced in comparing with that of the original. The block diagram of the multi-access memory system is presented in Figure 1.

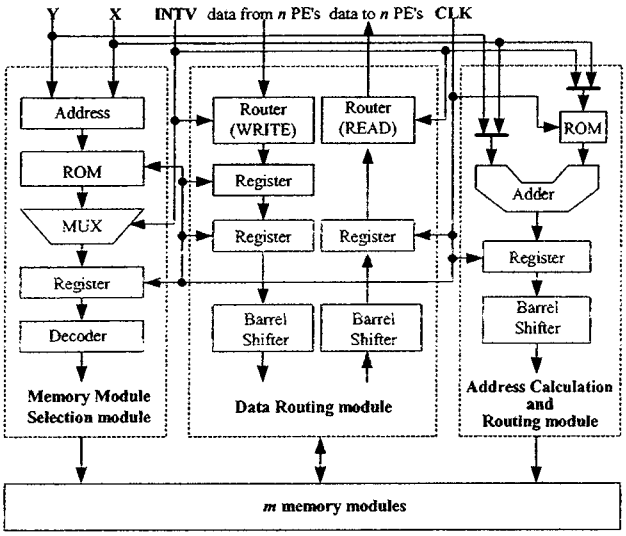

Figure 1. Functional Block Diagram of a Pipelined Multi-Access Memory System.

3. Pipelined Parallel Processing System

To perform high-speed visual applications, we had been developed the variety of parallel processing systems using MAMS. But, the performance of these systems did not give us faithful speed because two modules, which are MAMS module and the modules including processing elements, are asynchronous[4,5]. That is, when a memory instruction followed by a general instruction in the early developed parallel processing system, the general instruction had to wait to be excuted until the previous memory instruction was done completely. Although these systems give efficient fuctionality for accessing data in logical 2-D memory array simultaneously, memory access time in them is longer than one of general memory system, which does not support parallelism, because these system were depended on the time going through MAMS. It was main drawback to grow the processing speed up to the limitation of Amdal's law. To improve the processing speed of these systems, we have been investigated a pipelined parallel processing system together with MAMS.

The pipelined parallel processing system we proposed consists of a processing unit that is made up of a processor module(Motorola MPC860P Processor) for global processing and controling all devices and a PCI Controller (PLX9056) for transferring data to/from host computer; a local memory which stores instructions and common data for its programmability; DMA controller which has set of registers to fetch instructions from the local memory and issue them to n PEs; PE which interprets and executes instructions synchronously; a multi-access memory controller (MAMC) which provides n data elements to n PEs simultaneously; and m external memory modules which store n data elements to be manipulated.

The processor unit(PU) controls the system and communicates with host computer via PCI bus. DMA controller fetches an instruction and stores it in a register pool also synchronously transfers data or an instruction to n PEs and controls them. PE was designed as ALU with the functionality that is interpreting an issued instruction. And, to perform an application, DMA controller steels bus cycles until the end of the application.

PE can execute two kinds of instructions: memory-reference instructions for accessing m external memory modules via MAMC and 16 general instructions including register-reference instructions and I/O instructions. Therefore, an application to be processed on the system is compiled to operation codes within 18 instructions. And, when each of two instructions is in different set of instructions, they are executed at the same time. That is, one of memory-reference instructions and one of general instructions are executed simultaneously. Hence, a memory-reference instruction followed by a general instruction is executed in a memory access cycle and vise versa. It is reducing processing time in some modules, for example, convolution mask operators which is frequently used in the spatial domain.

The system can provide logically two-dimensional addressing way, which is used in *(r,c)*-based image domain, to system programmers because MAMS gets away sementic gap. For this feature of the system, most of image processing in spatial domain are done with enough parallel processing power. The block diagram of the system is presented in Figure 2.

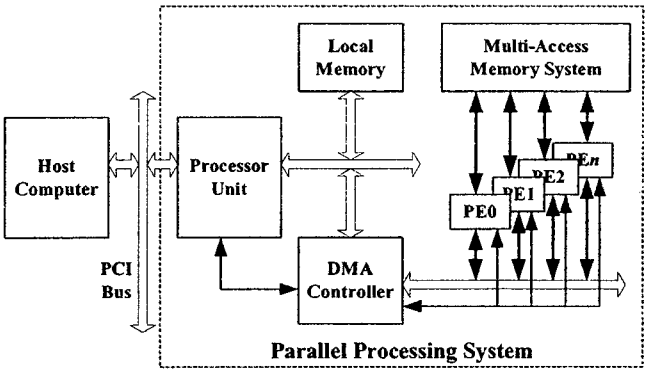

Figure 2. Functional Block Diagram of the Proposed Parallel Processing System.

4. Experiments

To verify the performance of the proposed system, we chose one of convolution mask operator that is used in image processing methods frequently but it is one of time consuming works.

We tried to transform the codes of the operator into adaptable codes to the proposed system in order to achieve the parallel version of this operation. The codes for the operation are presented in Code 1. Notice that two instructions, one is of memory-reference instructions and

the other is of general instructions, are executed simultaneously.

Code 1.

Read (1,0,0)	ValTran AC 0
Read (1,1,0)	Add rd_reg
Read (1,2,0)	Add rd_reg
...	...
NOP	Add rd_reg
NOP	Div rd_reg
Write(1,1,1)	Nop

The dedicated parallel processing system was described by Verilog-HDL and simulated by CADENCE Verilog-XL hardware simulation package in order to verify its functionality. And the system was compiled and fitted into EFP10K200EGC599-1 device by MAXPLUX II. The final simulation was performed after getting delay files and doing back-annotation. The waveform generated during the simulation is illustrated in Figure 3.

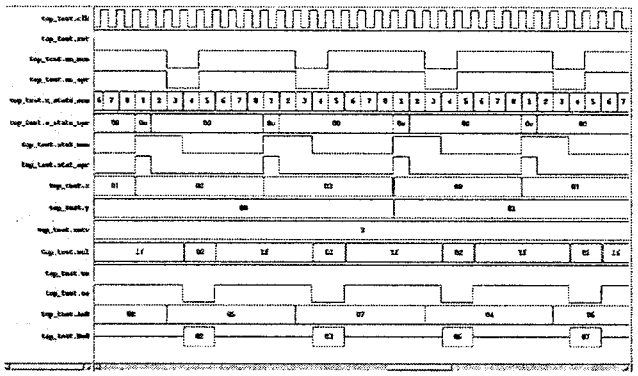

Figure 3. A Waveform obtained through post-layout simulation.

The waveform shows that fetching an instruction, one of memory access instructions, and one of arithmetic logic instructions were overlapped. And, we know that execution time of an application applied to the proposed system was depended on the memory access time. In the simulation results, the proposed system yielded 2.5 times faster speed than the early developed systems[4,5].

Unfortunately, we can not present real measuring values because the circuit board manufactured has not yet been verified. The work is going right now and the test environments are depicted in Figure 4.

5. Conclusions

The demands for processing multimedia data in real-time using unified and scalable architecture are ever increasing with the proliferation of multimedia applications. We had been developing the variety of parallel processing systems in order to improve the processing speed of visual media applications. These systems were using MAMS as a parallel memory system, which provides the capability of the simultaneous accesses of image points in a line-segment with an arbitrary degree. But, the performance of these systems did not give a faithful speed because of asynchronous feature between MAMS and processing elements.

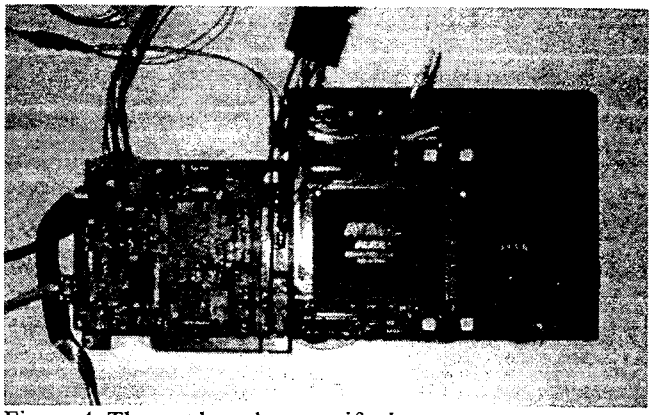

Figure 4. The test board : to verify the system.

To improve the processing speed of these systems, we have been investigated a pipelined parallel processing system using MAMS. Although the system is considered as being the SIMD type like the early developed systems, the performance of the system yielded 2.5 times faster speed.

Although the comparison values previously mentioned were obtained through simulations and speedup performances during each application on the system were achieved, they were just estimated values because the proposed system has not yet been verified on the circuit board manufactured.

Unfortunately, some problems are yet occurred in transferring a lot of data elements from the host to the system and vise versa during processing the application on the system. That is, the time for transferring data allocated more than the processing time. To solve this, the bus bandwidth needs to be improved on the system side and new specific methods to the system be developed on the method side.

References

[1] Alexander C. P. Loui, et. al, "Flexible Architecture for Morphological Image Processing and Analysis," *IEEE Transaction on Circuits and Systems for Video Technology*, Vol. 2, no. 1, Mar. 1992

[2] J. W. Park, "An efficient memory system for image processing," *IEEE transactions on Computers*, vol. C-35, no. 7, pp.33-39, 1986.

[3] J. W. Park and D. T. Harper III, "an Efficient Memory System for SIMD Construction of a Gaussian Pyramid," *IEEE Transactions on Parallel and Distributed System*, Vol. 7, No. 7 July 1996.

[4] Hyung Lee, K. A. Moon, J. W. Park, "Design of parallel processing system for facial image retrieval", *4th International ACPC'99*, Salzburg, Austria, Feb. 1999.

[5] Hyung Lee, J. W. Park, "A study on Parallel Processing System for Automatic Segmentation of Moving Object in Image Sequences," *ITS-CSCC 2000*, Vol. 2, pp. 429-432, July 2000.