Invited Paper

The SIGMA-1 Dataflow Supercomputer: A Challenge for New Generation Supercomputing Systems

KEI HIRAKI, KENJI NISHIDA, SATOSHI SEKIGUCHI, TOSHIO SHIMADA and TOSHITSUGU YUBA

SIGMA-1 is an exciting practical computer prototype based on dataflow architecture designed to outperform conventional von Neumann computers. This large-scale dataflow computer, for scientific and technological computations, is in the final design and construction stage at the Electrotechnical Laboratory, Ministry of International Trade and Industry.

This paper[1] overviews the system architecture of the SIGMA-1, and discusses the performance measurements made recently. The architectural issues in dataflow computing machines and the design decisions made in the SIGMA-1 project in achieving 100 MFLOPS performance are presented. The history of the SIGMA-1 project is also outlined.

1. Introduction

Supercomputers with the highest computation speeds are used in various numerical simulation applications, where experimental studies are unsafe and/or difficult. The computing power of supercomputers has increased several orders of magnitude during the last decade. This has been accomplished by pipelined vector architectures and multiple pipeline architectures. Recently, the multiprocessor architecture has been adopted to increase the performance further. However, its performance is limited severely by such large overheads as load distribution, synchronization and data transmission. Conventional supercomputers, consequently, only intend to utilize the coarse-grain parallelism, and neglect the enormous fine-grain parallelism available in numerical computations.

Dataflow computing promises computing speeds limited only by data dependencies in the computations being performed. Dataflow computing gives the simplest and most powerful representation of computation. Dataflow exploits parallelism in parallel computations at the architectural level with small overhead. The load distribution is dynamically achieved by an adaptive load scheduling mechanism built in the communication network. The synchronization of parallel activities is realized by efficient operand matching and asynchronous array access, while the data transmission between processing elements is done in pipeline manner.

The concepts of dataflow computers as they are now widely understood originated in 1974. In that year, two important papers on dataflow computer architecture were published, Jack B. Dennis's [3] and Raymond E. Miller and John Cocke's [9]. The former gave the dataflow architecture concepts beginning with the data dependency graph model of parallel computation. Then, it is possible to say that the foundation was made by the Dennis' group at MIT. Miller and Cocke of IBM proposed configurable computers as a new class of computers in the latter paper. They insisted that the machine structure should attain the natural structure of the algorithm being performed [9]. After these, the contributions to dataflow computer architecture were made by Arvind's group [1] at MIT and Gurd's group [4] at Manchester University to the present level of development of dataflow computers. Significant contributions have been made by many other researchers at many other institutions in Japan [11]. Much research and development went into implementing the dataflow computing concept at the hardware level using conventional microprocessors or constructing small scale experimental prototypes. However, none of these prototypes prove that dataflow computers can surpass conventional vector-type supercomputers. Quantitative analysis is needed to show the superiority of the dataflow architecture in practical supercomputing applications.

The SIGMA-1 is the most significant dataflow research project, undertaken by the computer architecture section of Electrotechnical Laboratory (ETL) to outperform conventional supercomputers. In addition to building such a powerful, new generation computer

Electrotechnical Laboratory, 1-1-4, Umesono, Tsukuba, Ibaraki 305, Japan.

[1]Note: This paper is a revision of our previous work on the subject: "System architecture of a dataflow supercomputer," Proc. 1987 IEEE Region 10 Conference, Seoul, Korea and "The SIGMA-1 dataflow computer," Proc. 1987 Fall Joint Computer Conference, Dallas, USA.

and verifying the feasibility of a dataflow computing machine as a practical parallel computer, the ETL group was motivated by the following.

1. The newest vector-type supercomputers have maximum performance of more than 1 GFLOPS now. However, their effective speed varies from 10 to 100 MFLOPS, depending on the application programs. This motivated the development of a general-purpose parallel computer to reduce the difference between peak performance and effective performance.

2. Semiconductor technology innovations have not been reflected fully in basic computer architecture research. Therefore, advanced research and development on new generation computer architectures must be conducted by utilizing advanced semiconductor technology. This motivated the development of a highly parallel computer to make full use of gate-array LSI technology.

The goals of the SIGMA-1 project include constructing a large-scale parallel computer with instruction level dataflow processing elements using advanced, commercially available, semi-customized (gate-array) LSI technology to execute practical application programs at an average speed of 100 MFLOPS and to build programming environments to support large-scale parallel computations.

To achieve these goals, a number of important design choices have been made in the SIGMA-1 [5, 6, 7, 10]. Two-stage pipeline processing is employed to improve the efficiency of executing sequential code segments. A new chained hashing hardware method is used to implement the operand matching mechanism. A hierarchical network is used for packet communication, where a new dynamic load distribution mechanism is implemented. A maintenance architecture for testing and debugging purposes and structure elements to support array structures with asynchronous access are introduced.

As mentioned above, the higher computing speeds expected in parallel computers with the increasing number of processing elements are limited by the parallel processing overhead. Dataflow computers are no exception to this phenomenon. Overheads include packet communication (data transmission), load distribution, packet waiting and matching (synchronization). To overcome these overheads, a quantitative evaluation on a practical dataflow computer is essential.

There is another potential overhead in dataflow computers. That is, a large number of extra instructions must be executed to preserve the safety of dataflow computations [4]. Additional instructions are necessary to maintain functional units such as the matching memory and structure memory in a dataflow computer. All this increases the complexity of the dataflow overhead problem in dataflow computing. The design of an efficient instruction set for a dataflow computer is an important research topic. To design an optimized instruction set, practical programs must be executed on a working dataflow computer. In order to solve these dataflow overhead problems, at least partially, it is essential to construct a practical dataflow computer and analyze its performance characteristics.

2. Hardware Architecture

2.1 Organization

The global organization of the SIGMA-1 is shown in Figure 1. The SIGMA-1 consists of 128 processing elements, 128 structure elements, 32 local networks, a global network, 16 maintenance microprocessors, a ser-

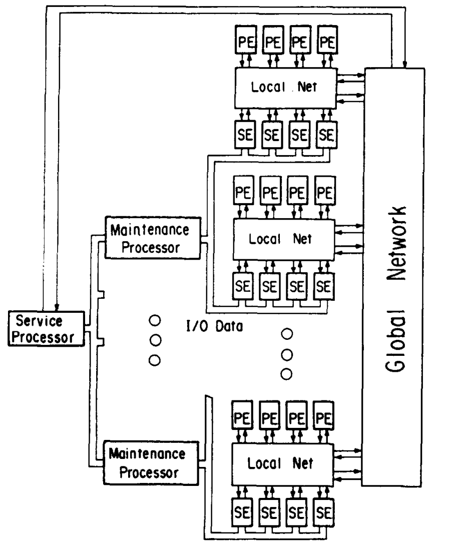

Fig. 1 Global organization of the SIGMA-1.

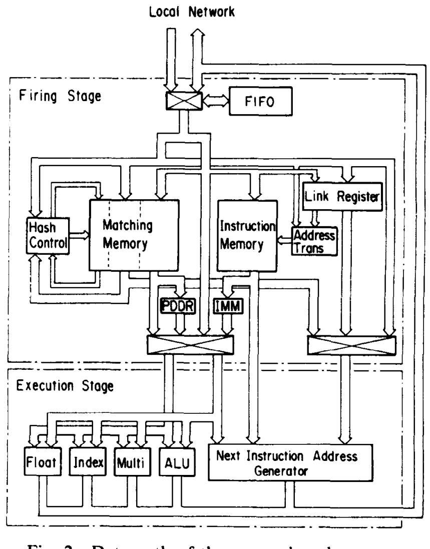

Fig. 2 Data path of the processing element.

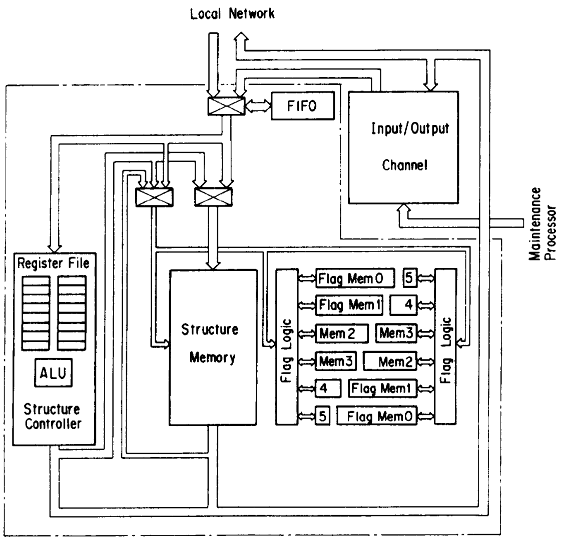

Fig. 3 Data path of the structure element.

vice processor and a host computer. The processing elements and structure elements are divided into 32 groups, each of which consists of four processing elements, four structure elements and a local network. All groups are connected via the global network. The global network consists of a two-stage omega network with a new adaptive load distribution mechanism. Its transfer rate is 2G bytes per second. A local network is a ten-by-ten crossbar packet switching network, eight ports of which are used for communication between processing elements and structure elements within a group and two to interface the global network. The transfer rate of a local network is 600M bytes per second. Input and output channels in all the structure elements, maintenance processors and a service processor form the maintenance architecture. This architecture is used for input and output operations, system monitoring, maintenance operations and performance measurements. The whole system is synchronous and operates under a single clock.

Figure 2 and Figure 3 illustrate a processing element and a structure element. A processing element consists of an input buffer, a matching memory with a waiting-matching function, an instruction memory and a link register file. A processing element executes all the SIGMA-1 instructions except structure handling instructions such as **read, white, allocate,** and **deallocate** used to access the structure memory in a structure element. A processing element works as a two-stage pipeline; the firing stage and the execution stage. In the firing stage, operand matching and instruction fetching are performed simultaneously. Input packets, stored in the input buffer, are sent to the matching memory. The matching memory enables the execution of instructions with two input operands. Successfully matched packets are sent to the execution stage with the instructions fetched from the instruction memory. If the match fails, the packet is stored in the matching memory and the instruction fetched is discarded. Chained hashing hardware is used to speed up the operand matching operation, where a first-in-first-out queue is attached to each key to enable multiple entries with the same key. The link register is used to identify a parallel activity in a program and hold a base address of a procedure.

In the execution stage, instructions are executed in parallel with packet identifier generation, where a packet identifier keeps the destination address (next instruction address) as well as a parallel activity identifier (a procedure identifier and an iteration counter). These operations are similar to those in a conventional computer. After combining the result value of the execution unit with the packet identifier, an output packet is produced and sent to the input buffer of the same or another processing element via the local network.

If an array is treated as a set of scalar values, the matching memory provides automatic synchronization and space management. However, heavy packet traffic causes a serious bottleneck at the matching memory, since each element of the array must be handled separately. This is the reason additional structure elements are introduced in the SIGMA-1. A structure element consists of an input buffer, a structure memory, two flag logic functions and a structure controller with a register file. The flag logic function is capable of test-and-setting all flags in an area arbitrarily specified in a single clock. The structure controller is responsible for waiting queue control and memory management. A structure element handles arrays, which are the most important data structure involved in numerical computations. Each structure cell is composed of a 40-bit data word and two flag bits used for **read** and **write** synchronization. Multiple **read** before **write** operations are realized by a waiting queue attached to each cell. The buddy system implemented in microprogramming is used to increase the speed of allocating and deallocating structures.

The architectural features of the SIGMA-1 are designed to achieve high speed by decreasing the parallel processing overheads in dataflow computers. For example, the short pipeline architecture enables quick response to small sized programs with low parallelism [5]. It was anticipated that the matching memory and the communication network would play a significant role in the system design from the viewpoint of synchronization and data transmission overhead. Besides the newly proposed mechanisms, high-speed and compact gate-array LSI elements have been developed for this purpose.

Testing, debugging and maintenance are performed by a special purpose parallel architecture [6]. This maintenance architecture uses a set of conventional von Neumann microprocessors, since maintenance operations generally need history sensitivity utilizing side effects.

The architectural features of the SIGMA-1 are summarized as follows:

1. A short (two-stage) pipeline in a processing element, reducing the gap between the maximum and average performance.
2. Chained hashing hardware for the matching memory, giving efficient and high-speed synchronization.
3. An array-oriented structure element, minimizing structure handling overhead.
4. A hierarchical network with a dynamic load distribution function, reducing performance degradation caused by load imbalance.
5. A maintenance architecture for testing and debugging, providing high-speed input and output operations and performing precise performance measurements.

2.2 Packet and Instruction Architecture

The SIGMA-1 adopts a packet communication architecture. Processing elements and structure elements communicate using fixed length packets. Input and output to the host computer and maintenance system are also in packet form. Therefore, hardware initialization and hardware maintenance are carried out in packet form.

As shown in Figure 4, a packet contains a destination processing element number, a packet identifier and tagged data, and miscellaneous control information. The 89-bit packet is divided into two 40-bit segments, a processing element number field (8 bits) and a cancel bit. A packet is transferred in two consecutive segments; the first segment contains a processing element or structure element number, a destination address in the processing element or structure element (32 bits), and packet type (8 bits). The second segment consists of data (32 bits), data type (8 bits) and one cancel bit. The cancel bit is used for canceling the preceding segment when a malfunction occurs. When the cancel bit is on, the whole packet becomes invalid.

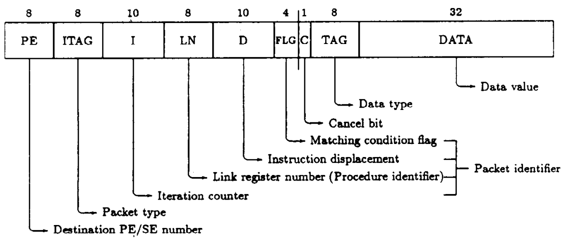

Fig. 4 Packet format.

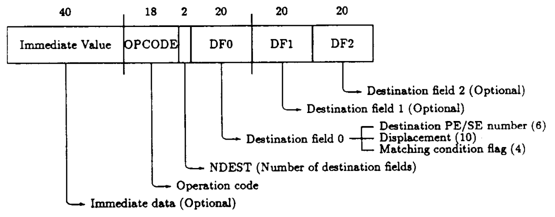
40
18
2
20
20
20
Immediate Value
OPCODE
DF0
DF1
DF2
Destination field 2 (Optional)
Destination field 1 (Optional)
Destination PE/SE number (6)
Displacement (10)
Matching condition flag (4)
Destination field 0
NDEST (Number of destination fields)
Operation code
Immediate data (Optional)

Fig. 5 Instruction format.

A destination address consists of a procedure identifier (8 bits) for specifying a parallel activity, a relative instruction address (10 bits) within the activity, an iteration counter (10 bits) and some control information determining the firing rule in the matching memory (4 bits). The iteration counter is utilized to implement the **loop** construct efficiently. As in the ordinary model of dynamic dataflow [1, 4], the concatenation of the procedure identifier and iteration counter is used to distinguish parallel activities in a program. The types of packets used are: result of instruction, procedure call and return, interrupt handling and system management, structure operation, and system initiation and maintenance for system resources.

The minimum length of an instruction is 40 bits (one word) as illustrated in Figure 5. The first 20 bits indicate the operation to be performed (18 bits) and the number of destination address fields (2 bits). The next 20 bits indicate the destination address of the result, which does not contain dynamically assigned information such as a procedure identifier and iteration counter. An immediate data operand can be located at the header part of each instruction. The maximum number of destination address fields is three, using two words. When an instruction contains an immediate data (constant) operand, another word is required.

The objective of the SIGMA-1 instruction set design is to execute efficiently application programs that cannot be efficiently executed on a vector-type supercomputer or a parallel von Neumann computer. Low efficiency in these program executions is caused by frequent procedure calls and returns, large amount of scalar arithmetic operations, and wide fluctuations of parallelism. Fine-grain parallelism has the advantage over coarse-grain parallelism to overcome this inefficiency. As a result, 200 instructions on a processing element and 97 instructions on a structure element have been implemented.

2.3 Hardware Implementation

The preliminary version of the processing element and structure element was built to confirm the architectural design of the SIGMA-1. The hardware technology used for the processing element was advanced Schottky TTL logic and MOS memories. The total number of ICs in the implementation is about 1,900. The AMD2900 series is also used for the structure element. The processing element consists of six printed circuit boards, and the structure element consists of a printed circuit board and a wire wrapping board. Each four layer printed circuit board is 40 cm × 60 cm (Fig. 6). The basic cycle is 80 ns for the processing element and 155 ns for the structure element. This prototype is con-

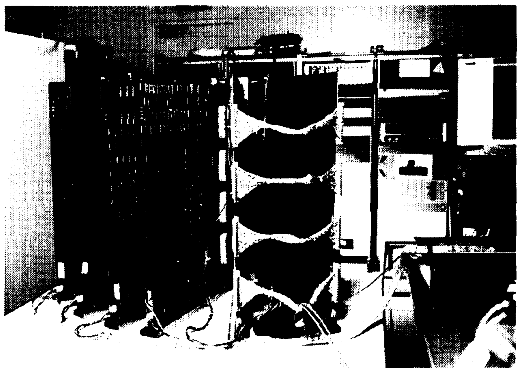

Fig. 6 Preliminary prototype of the processing element.

nected to a host computer, VAX-11/750, via a parallel interface.

LSI implementation using gate-array technology is generally used in commercial computers, but has not been used for developing experimental working prototype systems because of designing and manufacturing costs. In highly parallel computer development, customized and/or semi-customized LSI technology have a great advantage in physical size, power consumption and reliability.

The hardware technology chosen for the final implementation is widely used in the construction of commercial CMOS computers. The available gate-array LSI technology requires fewer printed circuit boards than currently used. The final version of both the processing element and structure element is constructed on a single printed circuit board, which consists of CMOS gate-array LSIs, CMOS memories, and SSIs for drivers and receivers. Each six layer printed circuit board is 45 cm × 45 cm. The size of the processing element is reduced approximately five times by using gate-array LSI technology.

High-speed arithmetic logic is effective in achieving high speeds in numerical computations. The AMD29325 was adopted for floating point arithmetic in each processing element, after fabricating a prototype using discrete ICs. The processing element has eight types of gate-array LSIs and a standard cell LSI, and contains 29 LSIs. The structure element contains eight LSIs. The numbers of logic gates in the processing element and structure element are approximately 81,000 and 37,000. The memory capacities of the processing element and structure element are about 1 M bytes and 1.5 M bytes. The total number of logic gates in the SIGMA-1 is about 19×10^6 and the total amount of memory is more than 300 M bytes. The basic cycle is 100 ns (10 MHz clock).

The ten-by-ten crossbar switch is built using four bit-slice ten-by-ten router chips, fabricated by using gate-array LSI technology at the 10,000 gate level. The router chips are adopted both in the local network and global network. The router LSIs for the local network

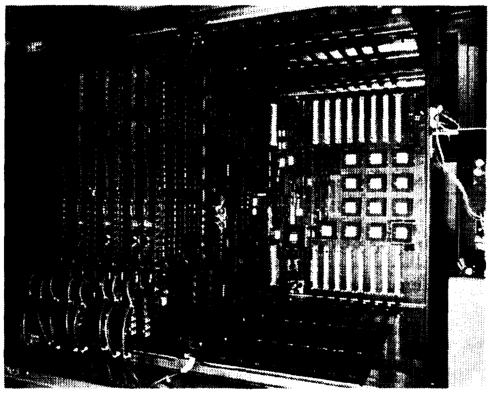

Fig. 7 One group organization of the SIGMA-1.

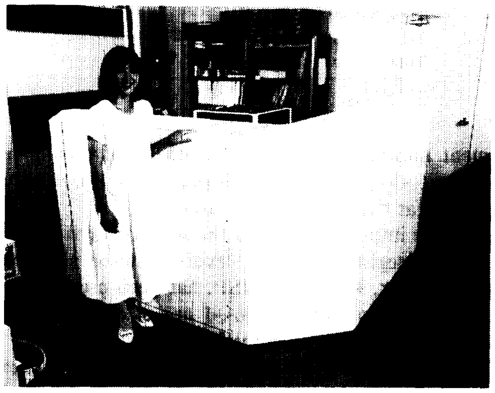

Fig. 8 Outside view of the SIGMA-1.

are directly installed in the backboard with a maintenance processor (Intel 8086), since there are several thousand wires between a local network and processing elements. Fig. 7 shows one group of the SIGMA-1 and the backboard, and Fig. 8 illustrates the external appearance of the whole system. The global network is placed in the center of the SIGMA-1 and connected to each local network by fine-pitch ribbon cables. The hardware specification is summarized in Table 1.

3. Software System

The SIGMA-1 system works as a back-end processor of a host computer. Therefore, it does not need a complicated operating system as in a stand-alone computing system. The SIGMA-1 has system programs for cleaning the input buffer, the matching memory, the structure memory and the instruction memory, as well as an initialization program for loading a program to be executed from the host computer into the instruction memory of each processing element. Output from the SIGMA-1 to the host computer is carried out in packet form by specifying the host computer as a processing

Table 1 Hardware specification of the SIGMA-1.

	Preliminary Version (Prototype)	Gate-Array Version (Final Configuration)
Technology:	Advanced STTL, CMOS SRAM	CMOS Gate-Array, SRAM, DRAM
Basic Clock:	155 ns	100 ns
No. of Gates:		81, 235/PE+37, 547/SE=118, 782
No. of ICs:	1570(PE)+320(SE)=1890	354/PE+353/SE=707
No. of PCBs:	6(PE)+2(SE)=8	1/PE+1/SE=2
No. of PE/SE:	1PE, 1SE	128PE, 128Se
Input Buffer:	2KW(80 bits/W)	8KW(80 bits/W)
Matching Memory:	16KW(80 bits/W)	64KW(80 bits/W)
Instruction Memory:	16KW(40 bits/W)	64KW(40 bits/W)
Structure Memory:	64KW(40 bits/W)	256KW(40 bits/W)
No. of Gate-Array Types:		9/PE, 3/SE, 2/LNET, 1/GNET
No. of Gate-Arrays Used:		28/PE, 8/SE, 13/LNET, 117/GNET
Total Amount of Memory:		326MByte
Total No. of Gates:		19, 013, 447
Total No. of ICs:		91, 029
Total No. of PCBs:		128(PE)+128(SE)+16(LNET)+9(GNET)=281

element number of a packet. Data input during program execution is given also in packet form by the host computer at the SIGMA-1's request.

Another important part of the software system is the SIGMA-1 register transfer level software simulator. The simulator, written in LISP, has been used to verify each function of the SIGMA-1 and is still utilized to debug assembly language test programs.

The Data Flow C (DFC) compiler is implemented using the language development tools of the UNIX operating system. The emphasis was not on efficiency but on ease of implementation. Efficiency and compactness are still to be considered. Basically, DFC is a subset of the C language with a single assignment rule. Therefore, assignment to a variable is generally allowed only once in each procedure. However, the **loop** construct, realized by the **for** statement, is an exception. A non-single assignment expression can be written as the third argument of the **for** statement as shown in the following summation program. The argument *n* of the main program is given by a trigger packet invoking the program.

```
#define N 10
main(n)
  int n;
  {int i; float a[N], retval, sigma( );
  for (i=0; i<5; i=i+1) a[i]=i;
  retval=sigma(n, a); print(retval);}
float sigma(n, a)
  int n; float a[ ];
  {int i; float s;
  for (i=0, s=0; i<n; s=s+a[i], i=i+1);
  return(s);}
```

DFC may have multiple return values as an extension to the the C language. DFC programs without multiple return values can be compiled by a C compiler. The advantage is that DFC programs can be verified by a C compiler on a host computer. The DFC compiler comprises approximately 7,000 steps in C. A nested structure of **for** or **if** statements is not implemented and is inhibited by the current version of the DFC compiler.

The DFC compiler generates a macro-assembly language program. This corresponds to a dataflow graph with no restriction on the number of arcs. Each node is presented by a lable, an operation code, constant, and a destination list, where the constant part might be absent. An element of a destination list consists of a processing element number, a statement label, and a matching condition flag. When a processing element number is not specified, it is dynamically allocated. Currently, it is our policy that a processing element number must be statically assigned at compile time, since dynamic allocation causes execution overhead. The matching condition flag contains characteristic information concerning the next instruction.

From now on, we will focus our software development efforts on the following:

1. Development of an efficient and robust DFC compiler for constructing large-scale practical programs,
2. Construction of reliable maintenance and programming environments,
3. Development of practical scientific application programs for evaluation.

4. System Performance

The whole system operates synchronously at a 10 MHz clock beat. The execution time of an instruction is determined by the maximum execution time of the two pipeline stages. Single operand instructions are executed every two cycles and two operand instructions every three cycles. The firing stage normally takes three cycles. Since non-division arithmetic operations take at most two cycles, the execution stage completes most of the instruction execution in two cycles. This consideration implies that the maximum speed of a processing element is about 5 MIPS and 3 MFLOPS. For a structure element, each **read** and **write** instruction is carried out in two cycles. Since instructions for allocating and deallocating structures are implemented by a microprogrammed buddy system, the execution times estimated are between 10 to 340 cycles. The network transfers a packet every two cycles. Therefore, structure elements can utilize maximum performance of the network.

Performance is to be measured in terms of program execution time rather than raw machine cycles. The measured performance of the preliminary version of the SIGMA-1 is approximately 1.4 MIPS. The FLOPS performance is approximately 0.3 MFLOPS, measured by executing the Lawrence Livermore Loops program which is not suited to dataflow architecture. This measurement study proved that there is no difference in computing ability of a single processing element between dataflow architecture and von Neumann architecture, in case of constructing them using the same device technology.

Performance evaluation on the four processing element SIGMA-1 of the LSI version was studied by executing benchmark programs. The program executed computes the numerical integration of **sin** (x) over $-\pi/2$ to $\pi/2$ using the trapezoidal method. The number of divisions on the integration range corresponds to the length of the vector. The program was written in DFC, and then optimized manually. The clock cycles were measured using a logic state analyzer.

The following observations were made in this study:

1. The effective performance of the four processing element organization is 6.4 MFLOPS, approximately half of the maximum performance. The speed of the FACOM M380, a general-purpose mainframe computer, was 9.6 MFLOPS for this benchmark.
2. A speed degradation of 30% occurs when structure elements are adopted. This is due to the extra instructions and address calculation for structure handling.
3. The single processing element performance is almost constant over the vector length, because of the short pipeline architecture.
4. The speedup ratio for a single processing element to four processing element organization is 3.9 without structure elements and 3.5 with structure elements.

This preliminary performance evaluation predicts that the average performance of the 128 processing element organization is more than 100 MFLOPS.

5. History

The SIGMA-1 project started in April 1982 as part of the national project for new generation supercomputing systems under the auspices of the Ministry of International Trade and Industry, Japan. The national project aims at research and development of the basic technology for next generation supercomputers based on parallel processing with new devices. The main objective of the SIGMA-1 project is to identify and establish the basic technology for highly parallel dataflow supercomputers and verify the ability to surpass successful conventional von Neumann computers by a practical dataflow computer.

In 1982, the proposal for the SIGMA-1 architecture was outlined and its software simulator, written in LISP, was implemented on the DEC20/60 and the LMI's Lambda Machine. The simulation study conceptually verified the dataflow architecture. Hardware design started in 1983. A preliminary version of the processing element was fabricated and has been in operation since November 1984. A paper describing the architectural design of the SIGMA-1 was presented at the 1984 International Conference on Parallel Processing [5]. The performance evaluation of the preliminary prototype SIGMA-1 was carried out to confirm the effectiveness of the new control schemes and was reported at the International Symposium on Computer Architecture held in 1986 in Tokyo [10].

From 1984 to 1985, the SIGMA-1 processing element was redesigned using semi-customized (gate-array) LSI technology to reflect the features of the preliminary processing element. The architecture was presented at the 1986 International Conference on Parallel Processing [6]. A local communication network was implemented, and a single cluster SIGMA-1 consisting of four processing elements and a local network was constructed and has been in operation since December 1986.

The SIGMA-1, with 128 processing elements, will be completed in 1987. Software implementations effectively utilizing dataflow architecture and controlling and managing resources are planned after testing the hardware soundness. After establishing programming environments and confirming the usefulness, the SIGMA-1 will be in open use.

The man power used in this project may be of some interest. A group of four researchers was involved in the development, two for hardware and two for software. The design of the preliminary processing element was carried out by the staff at ETL using a non-commercial logic simulator. However, its fabrication was done by Central Electric Corp. The gate-array design of the LSI version of the processing element was carried out by Central Electric Corp. staff, with no experience of logic design, mainly using the HILO-2 logic simulator on a CAD workstation.

The research fund amounted to about 500 million yen, excluding salary spreading over six years. 70% of the fund has been used to fabricate the SIGMA-1 system. The remainder is for computing facilities, their maintenance charge, electricity, etc. The number of processing elements was greatly influenced by the financial support. Initially, we planned to construct a parallel computer consisting of more than 200 processing elements. However, the hardware prototype will only be a reduced version of the potentially large system that was designed.

6. Concluding Remarks

So far, no attempt has been made to maintain compatibility with conventional von Neumann computer architecture. From now on, we must evaluate the practicability of the SIGMA-1 by comparing its performance with von Neumann computers. In order to make a detailed comparison, a common high-level programming language is required. We have implemented a SISAL [8] compiler for the SIGMA-1 by using its intermediate language. Application programs in SISAL will be executed on the SIGMA-1, CRAY supercomputers and VAXs. This type of compatibility is required to show the superiority of dataflow architecture over von Neumann architecture.

The SIGMA-1 with 128 processing elements will be in operation at the end of 1987. This will mean that the center of dataflow research shifted away from the United States or England to Japan. The main reason is that our laboratory is a government owned organizations, where the basic researches requiring risk, a large amount of funds and long-range perspective are carried out. Research on dataflow computer architecture is a typical example satisfying such conditions. That is, industries are not interested in such revolutionary architecture and universities do not have enough research funds and technology to construct a practical dataflow computer.

Dataflow computer research is still at the feasibility study level, verifying the ability to surpass the successful conventional von Neumann computers by a practical dataflow computer. The SIGMA-1 dataflow computer, which is predicted to outperform conventional supercomputers in some applications, is an important mile stone in computer architecture research. Many evaluation studies will be carried out on the SIGMA-1 to confirm the effectiveness of dataflow computer architecture.

Acknowledgements

The authors wish to thank the staff of the Computer Architecture Section, Electrotechnical Laboratory, for their fruitful discussions. They are also grateful to Dr. H. Kashiwagi, Deputy Director-General of Electrotechnical Laboratory, for his continuous encouragement and Dr. Wim Bohm, the University of Manchester, for improving the manuscript. This research is supported by the High-Speed Computing System for Scientific and Technological Use Project of the Ministry of International Trade and Industry, Japan.

References

1. Arvind and Iannucci, R. A. A critique of multiprocessing von Neumann style, *Proc. 10th Annu. Int. Symp. Computer Architecture, IEEE*, 426-436 (1983).
2. Dennis, J. B., Lim, W. Y. P. and Ackerman, W. B. The MIT data flow engineering model, *Proc. IFIP Congress 83*, 553-560 (1983).
3. Dennis, J. B. First version of a data flow procedure language, Lecture Notes in *Computer Science*, **19**, Springer-Verlag, 362-376 (1974).
4. Gurd, J., Kirkham, C. C. and Watson, I. The Manchester prototype dataflow computer, *Commun. ACM*, **21**, 1, 34-52 (1985).
5. Hiraki, K., Shimada, T. and Nishida, K. A hardware design of the SIGMA-1—A data flow computer for scientific computations, *Proc. 1984 Int. Conf. Parallel Processing, IEEE*, 524-531 (1984).
6. Hiraki, K., Nishida, K., Sekiguchi, S. and Shimada, T. Maintenance architecture and its LSI implementation of a dataflow computer with a large number of processors, *Proc. Int. Conf. Parallel Processing, IEEE*, 584-591 (1986).
7. Hiraki, K., Sekiguchi, S. and Shimada, T. System architecture of a dataflow supercomputer, *Proc.* 1987 *IEEE Region 10 Conf., IEEE*, 1044-1049 (1987).
8. McGraw, J. SISAL: Streams and iteration in a single assignment language, Language Reference Manual, Lawrence Livermore National Laboratory (1985).
9. Miller, R. E. and Cocke, J. Configurable computers: A new class of general purpose machines, Lecture Notes in *Computer Science*, **5**, Spinger-Verlag, 285-298 (1974).
10. Shimada, T., Hiraki, K., Sekiguchi, S. and Nishida, K. Evaluation of a single processor of a prototype data flow computer SIGMA-1 for scientific computations, *Proc. 12th Ann. Int. Symp. Computer Architecture, IEEE*, 226-234 (1986).
11. Yuba, T. Research and development efforts on dataflow computer architecture in Japan, *J. Inf. Process.* 9, 2, IPS Japan, 51-60 (1986).

(Received November 17, 1987)