International Journal of Engineering Sciences Paradigms and Researches (IJESPR)
(Vol. 36, Issue 01) and (Publishing Month: November 2016)
(An Indexed, Referred and Impact Factor Journal)
ISSN: 2319-6564
www.ijesonline.com

An Application System for Image Processing and Pattern Recognition

Bhagaban Sri Ramakrishna

Aryan Institute of Engineering & Technology, Bhubaneswar, Odisha

Abstract— PASM, an enormous scope multi microprocessor framework being planned at Purdue University for picture handling and example rec-ognition, is portrayed. This framework can be powerfully reconfigured to work as at least one autonomous SIMD and additionally MIMD machines. PASM comprises of an equal calculation unit, wiiich contains N processors, N recollections, and an interconnection organization; Q miniature regulators, every one of which controls NJQ processors; NfQ equal optional stockpiling gadgets; an appropriated memory the executives framework; and a framework control unit, to arrange the other framework parts. Potential qualities for N and Q are 1024 and 16, individually. The control plans and memory the board in PASM are investigated. Instances of how PASM can be utilized to perform picture preparing assignments are given.

***Index* terms—Image processing, memory management, MIMD machines, multimicroprocessor systems, multiple-SIMD machines, parallel processing, partitionable computer systems, PASM, reconfigurable computer systems, SIMD machines.**

INTRODUCTION

As a result of the microprocessor revolution, it is now feasible to build multi microprocessor systems capable of performing image processing tasks more rapidly than previously possible. There are many image processing tasks which can be performed on a parallel processing system, but are prohibitively expensive to perform on a conventional computer system due to the large amount of time required to do the tasks (37]. In addition, a multi microprocessor system can use parallelism to perform the real-time image processing required for such applications as robot (machine) vision, automatic guidance of air and spacecraft, and air traffic control.

There are several types of parallel processing systems. An *SIMD* ***(single*** *instruction stream—multiple data stream) machine* [18] typically consists of a set of *N* processors, *N* memories, an interconnection network, and a control unit (e.g., Illiac IV [10]). The control unit broadcasts instructions to the processors and all active ("turned on") processors execute the same iRstruction at the same time. Each processor executes instructions using data taken from a memory with which only it is associated. The interconnection network allows inter- **processor** communication. An *MSIMD (itiultiple-SIMD) system* is a parallel processing system which can be structured as one or more independent SIMD machines (e.g., MAP [31], [32]). The Illiac IV was originally designed as an MSIMD system [3]. An *MIMD (multiple* ***instruction*** *stream-multiple data stream) machine* [18] typically consists of *N* processors and *N* memories, where each processor can follow an independent instruction stream (e.g., C.mmp [60]). As with SIMD architectures, there is a multiple data stream and an interconnection network. A *partitionable SIMDfMIMD system* is a parallel processing system which can be structured as one or more independent SIMD and/or MIMD machines. In this paper PASM [47J, [48], a partitionable SIMD/MIMD system being designed at Purdue University for image processing and pattern recognition, is described.

Many designers have discussed the possibilities of building large-scale parallel processing systems, employing 2^{1}* to $2'^{6}$ microprocessors, in SIMD (e.g., binary ii-cube array [34]) and MIMD (e.g., CHoPP [54], [55]) configurations. Without the presence of such a large number of processors, the concept of partitioning the system into smaller machines which can operate as SIMD or MIMD machines was unnecessary. Nutt [31] has suggested a machine which is a multiple-SIMD system. Lipovski and Tripathi [27] have considered the idea of combining the SIMD and MIMD modes of operation in one system. In addition, developments in recent years have shown the importance of parallelism to image processing, using both cellular logic arrays (e.g., CLIP [50], BASE 8 [35]) and SIMD systems (e.g., STARAN [36]). A variety of such systems are discussed in [19]. Thus, the time seems right to investigate how to construct a computer system such as PASM: a machine which can be dynamically reconfigured as one or more SIMD and/or MIMD machines, optimized for a variety *of* important image processing and pattern recognition tasks.

International Journal of Engineering Sciences Paradigms and Researches (IJESPR)
(Vol. 36, Issue 01) and (Publishing Month: November 2016)
(An Indexed, Referred and Impact Factor Journal)
ISSN: 2319-6564
www.ijesonline.com

The use of parallel processing in image processing has been limited in the past due to cost constraints. Most systems used small numbers of processors (e.g., Illiac IV [10]), processors of limited capabilities (e.g., STARAN [36]), or specialized logic modules (e.g., PPM [24]). With the development of microprocessors and related technologies it is reasonable to consider parallel systems using a large number of complete processors.

SIMD machines can be used for "local" processing of segments of images in parallel. For example, the image can be segmented and each processor assigned a segment. Then following the same set of instructions, such tasks as line thinning, threshold dependent operations, and gap filling can be done in parallel for all segments of the image simultaneously. Also in SIMD mode, matrix arithmetic used for such tasks as statistical pattern recognition can be done efficiently. MIMD machines can be used to perform different "global" pattern recognition tasks in parallel, using multiple copies of the image or one or more shared copies. For example, in cases where the goal is to locate two or more distinct objects in an image, each object can be assigned a processor or set of processors to search for it. An SIMD/MIMD application might involve using the same set of microprocessors for preprocessing an image in SIMD mode and then doing a pattern recognition task in MIMD mode.

PASM, a partitionable SIMD/3fIMD machine, is a large-scale dynamically reconfigurable multimicroprocessor system [43], [46]—[48]. It is a special purpose system being designed to exploit the parallelism of image processing and pattern recognition tasks. It can also be applied to related areas such as speech processing and biomedical signal processing. In this paper the architecture of PASM is presented and examples of its use in performing image processing tasks are given.

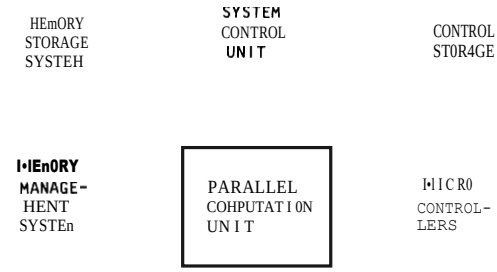
Computer system designers have been considering various multimicrocomputer architectures, such as [9], [11], [23], [26], [27], [34], [54], [58], and [59]. PASM combines the following features:

l) it can be partitioned to operate as many independent SIMD and/or MIMD machines of varying sizes, and

2) it is being developed using a variety of problems in image processing and pattern recognition to guide the design choices.

The purpose of PASM is to serve as a vehicle for experimenting with parallel image processing and pattern recognition. It is not meant to be a production-line machine, but rather a research tool. The design attempts to incorporate the needed flexibility for studying large-scale SIMD and MIMD parallelism, while keeping system costs "reasonable."

In this paper the overall organization of PASM is presented. In particular, the control structure and secondary storage systems are described, and some application examples are



given. The purpose here is to present design concepts that will allow a system to exploit the parallelism that, for example, 1024 processors can provide. Implementation details are currently under study and are beyond the scope and length of this paper.

Fig. l is a block diagram of the basic components of PASM. The *parallel comRutation unit (PCU)* contains *N -- 2•* pro-

Fig. 1. Block diagram overview of PASM.

cesors, *N* memory modules, and an interconnection network. The *PCU processors* are microprocessors that perform the actual SIMD and MIMD computations. The *PCU memory modules* are used by the PCU processors for data storage in **SIMD** mode and both data and instruction storage in MIMD mode. The *interconnection network* provides a means of communication among the PCU processors and memory modules. Two possible ways to organize the PCU and different types of partitionable networks which can be used are described in Section II.

The *microcontrollers {MC's)* are a set *of* microprocessors which act as the control units for the PCU processors **in SIMD** mode and orchestrate the activities of the PCU processors in MIMD mode. There are *Q -- 2** MC's. Each MC controls *NfQ* PCU processors. A virtual **SIMD** machine (partition) of size IV/Q, where A = 2* and 0 ñ *r S q,* is obtained by loading A MC memory modules with the same instructions simulta- neously. Similarly, a virtual **MIMD** machine of size *RN/Q* is obtained by combining the efforts of the PCU processors of A MC's. Q is therefore the maximum number of partitions allowable, and *NfQ* is the size of the smallest partition. Possible values for *N* and *Q* are 1024 and 16, respectively. *Control storage* contains the programs for the MC's. The MC's are discussed in more detail in Section **III.**

The *memory storage system* provides secondary storage **space** for the data files in **SIMD** mode, and for the data and program files in MIMD mode. The *memory management system* controls the transferring of files between the memory storage system and the PCU memory modules. It employs a set of cooperating dedicated microprocessors. Multiple storage devices are used in the memory storage system to allow parallel data transfers. The secondary memory system is described in Section IV.

The *system control unit* is a conventional machine, such as

a PDP-11, and is responsible for the overall coordination of the activities of the other components of PASM. Examples of the tasks the system control unit will perform include program development, job scheduling, and coordination of the loading of the PCU memory modules from the memory storage system with the loading of the MC memory modules from control storage. By carefully choosing which tasks should be assigned to the system control unit and which should be assigned to other system components, the system control unit can work effectively and not become a bottleneck. Examples of this in- clude using the MC's to act as the control units for virtual SIMD machines, controlling the interconnection network with routing tags generated by each PCU processor, and having the memory management system supervise primary/secondary memory transfers.

Together, Sections II, III, and IV present the overall architecture of PASM. Particular attention is paid to the ways in which the control structure and secondary memory scheme allow PASM to be efficiently partitioned into independent virtual machines. Variations in the design of PASM's PCU which still support these control and secondary memory ideas are examined. This examination demonstrates how the concepts underlying PASM can be used in the design of different systems.

In Section V image processing algorithms using PASM are presented. In particular, smoothing and histogram calculations are examined. Using these examples, the potential improvement a system such as PASM can provide over serial machines is demonstrated.

PARALLEL COMPUTATION **UNIT**

A. PCU Organization

The parallel computation unit (PCU) contains processors, memories, and an interconnection network. One configuration of these components is to connect a memory module to each processor to form a processor—memory pair called a *processing element {PE)* (e.g., Illiac IV [10]). The interconnection network is used for communications among PE's. This *"PE-to-PE"* configuration is shown in Fig. 2. A pair of memory units is used for each memory module. This double-buffering scheme allows data to be moved between one memory unit and secondary storage (the memory storage system) while the processor operates on data in the other memory unit. In the PE-to-PE configuration, "local" memory references are relatively fast, however, the transfer of large blocks of data from PE to PE is delayed by the memory fetching and storing which must be done.

The *"P-to-M"* (processor-to-memory) configuration, shown in Fig. 3, uses the interconnection network to connect the processors to the memory modules. Again, double-buffering is employed. There is no "local" memory. To fetch an operand from memory, the processor must first send the address of the operand through the interconnection network to a memory. Then the processor receives the operand from the memory via the interconnection network. Advantages of the P-to-M configuration are that a memory connected to a processor can be reconnected to another processor, effectively transferring the entire contents of the memory from one processor to another, and that the number of memories does not have to be equal to the number of processors (e.g., BSP [12]). A disadvantage is that all memory references must go through the interconnection network.

A more detailed analysis reveals some of the tradeoffs involved in these two ci›nfigurations. If Tmr US the time required for a memory access (either a read or a write), and *T;p* is the time to send a data item through the interconnection network, then the time required for a memory reference in the P-to-M

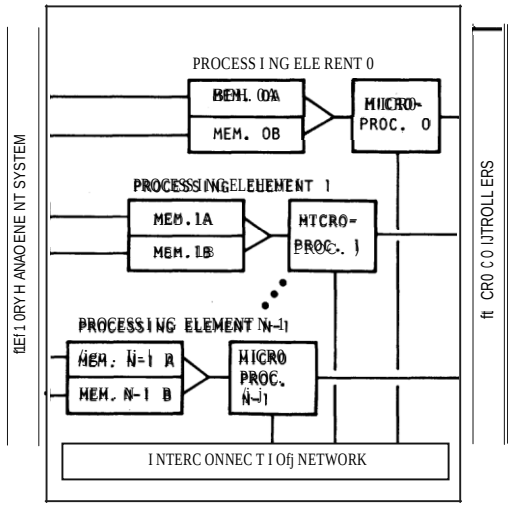


Fig. 2. PE-to-PE conriguration of the parallel computation unit.

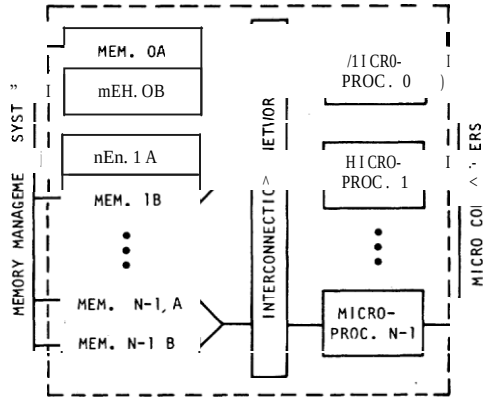


Fig. 3. Processor-to-memory configuration of the parallel computation

address to the memory and once for transferring the data. (The time required for controlling the network is omitted since control methods vary.)

For the PE-to-PE configuration the time required for a memory reference, F E. depends on the location of the memory which is to be used. If the memory is local, then

$$T_{pp} - T_z . \qquad (2)$$

If the memory is connected to some other processor, then configuration, r @- M. is given by

$$F \quad —M' \quad T_{in} + T_{mr} + \quad in• \qquad (l)$$

The must be included twice, once for the processor to send the

$$P_{PE'} \quad T_{jg} + T \quad p + F_{in} \bullet \qquad (3)$$

P_i represents the time required for the PE which has the data item to recognize and service the data request. This may re- quire a significantly longer delay than $T_{,,,,}$. Ifp is the proba- bility of a local memory reference, then (2) and (3) can be combined to give the expected memory reference time

$$E\ [\ T_{HE}]\ pT_{z,} + (1\ p)\ T_{ry} + T_f + T_{im} \bullet \qquad (4)$$

dress of the desired data in the processor connected to the memory to be accessed (e.g., see the algorithms in Section V-B). Thus, (4) reduces to

$$E\ T_{PE]'} \quad pT_{mr} + \quad 1 \quad p)\ _pTzg + T;\ )\bullet \qquad (6)$$

Therefore, when operating in SIMD mode the PE-to-PE configuration is preferable.

When operating in MIMD mode, the PE-to-PE configuration requires that two processors be involved in every non-local memory reference. The efforts of two processors involved in a data transfer can be coordinated by having the processor which initiates the transfer interrupt the other processor or by dedicating one of these processors to handling data transfers. In the P-to-M configuration the memories are shared by the processors, i.e., more than one processor can access the same memory for either data or instructions. However, for the image processing tasks that have been examined, most data and in- structions can be stored in the local memory, reducing the impact of this consideration.

The PE-to-PE configuration will be used in PASM. Depending on the application for which a different partitionable SIMD/MIMD system is intended, the P-to-M configuration may be preferable. The interconnection networks, control structure, and secondary memory system described below can be used in with either.

B. Interconnection Networks

Two types of multistage interconnection networks are being considered for PASM: the Generalized Cube and the Augmented Data Manipulator (ADM). Both of these networks have the following properties: 1) a logical structure consisting of n stages of N/2 (Cube) or *N* (ADM) switches [49]; 2) distributed control by routing tags generated by each processor [25], [28]; 3) the ability to operate in SIMD mode *(N* simul- taneous transfers) and MIMD mode [44], [45]; and 4) the ability to be partitioned into independent subnetworks [42]. As discussed in [49], the Cube network is equivalent or directly related to other networks in the literature (e.g., [5], [6], [25], [33], and [34]). The ADM is an extension of the data ma- nipulator [16].

Both the Cube and ADM networks can be implemented as single stage networks [38], [40], [41] instead of as multistage networks. These single stage implementations can also be partitioned [42], are compatible with the PASM control and memory management schemes, and may be appropriate for MSIMD systems, depending on the intended applications. However, since there is only a single stage of switches, for the MIMD mode of operation intermediate processors may have to be interrupted to transfer data. (See [40] for more infor- mation.) Thus, single stage networks are not appropriate for PASM, but might be for an MSIMD system based on the design **concepts of** PASM.

The tradeoffs between the Cube and ADM multistage networks for PASM are currently under study. The ADM network is more flexible and fault tolerant [40], [44], but is more complex. The Cube may be more cost effective and sufficient for the system's needs. In the following sections it will be assumed that the processors will be partitioned such that

Comparing (l) and (4)

$$T_{p-M} \ N E[T_{PEN} \qquad (5)$$

for *p* sufficiently large. Thus, the "best" configuration is task dependent.

When operating in SIMD mode with the PE-to-PE configuration, it is often possible to omit one occurrence of *The* in (3) and reduce F_I to *Tz,*.This is done by computing the ad-

the addresses of all of the processors in a partition agree in their low-order bit positions. This constraint will allow either the Cube or ADM network to be used as the partitionable inter- connection network in PASM. Details of these networks are beyond the scope of this paper, and readers are referred to the references indicated.

C. PCU Processors

The PCU processors will be specially designed for parallel image processing. Simulation studies are currently being conducted to aid in determining an efficient instruction set. A PASM prototype will most likely be based on user micro- programmable components to obtain needed flexibility, while the final system will employ custom VLSI processors.

IV. CONCLUSIONS

PASM, a large-scale partitionable SIMD/MIMD multimicroprocessor system for image processing and pattern recognition, has been presented. Its overall architecture was described and examples of how PASM can realize significant computational improvements over conventional systems was

International Journal of Engineering Sciences Paradigms and Researches (IJESPR)
(Vol. 36, Issue 01) and (Publishing Month: November 2016)
(An Indexed, Referred and Impact Factor Journal)
ISSN: 2319-6564
www.ijesonline.com

demonstrated.

PASM differs from other large-scale parallel processing systems whose capabilities include the ability to operate in SIMD mode in terms of its balance of flexibility and complexity. PASM is more complex than the CLIP4 [l 4], DAP [17], MPP [8], and BASE 8 processsors [35] in the sense that these systems use bit-serial processors connected in a four or eight nearest neighbor pattern. The PASM design is more flexible than the STARAN [4], [7], which uses bit serial processors and can only operate in the SIMD mode. The interconnection network in STARAN is a multistage Cube network, but it differs from the networks proposed for PASM in that it has a limited control scheme [5]. PASM is also more flexible than the Illiac IV [10], which has a four nearest neighbor interconnection network, and can only operate in SIMD mode. (The original Illiac design had four control units and could operate in MSIMD mode [3].) PASM differs from the BSP organization [52] in that the BSP uses 16 processors (a smaller scale of parallelism) and is limited to just the SIMD mode of operation. The MAP [3l] system is an MSIMD machine design in which the control units are connected to groups of processors via a crossbar switch, making its control unit to processor connection abilities *more* flexible than PASM's. However, the processors in MAP communicate with each other via a bus system, so all processors cannot send data in parallel, and there are no provisions for the system to operate in MIMD mode.

Two other microprocessor based systems capable of both MSIMD and MIMD modes are the Reconfigurable Varistructure Array Processor [27] (now called TRAC) and PM4 [111 TRAC does not have any explicit control units; SIMD processing is achieved by having more than one processor fetch instructions from the same memory. In addition, TRAC can combine the effects of processors to vary its machine word size. In PM4, the control units are connected to the processors via a crossbar type of switch, so any processor can be driven by any control unit. Also, in addition to a network for interprocessor communications, there is a different multistage network between the processors and shared memory. Another system capable of operating in the MSIMD and MIMD modes is the dc group system [22]. This is a dynamically reconfigurable system, capable of creating virtual machines of different word sizes. PASM is, in general, less complex and less flexible than each of these three systems. However, for the image processing tasks studied thus far, the PASM design has had sufficient flexibility.

In summary, the objective of the PASM design is to achieve a system which attains a compromise between flexibility and cost-effectiveness *for* a specific problem domain. Future work on PASM includes choosing a microprocessor architecture suitable for PASM, investigating system reliability and fault tolerance, designing an "intelligent" compiler for a high level parallel image processing language, specifying the hardware design details, and developing the operating system and programming languages for a prototype system. A dynamically reconfigurable system such as PASM should be a valuable tool for both image processing/pattern recognition and parallel processing research.

ACKNOWLEDGMENT

The authors would like to thank R. J. McMillen, G. B. Adams, III, P. A. England, J. Bogdanowicz, and M. Washburn for their contributions to the design of PASM, and L. Wittie for many useful conversations. They also gratefully acknowledge the comments of the referees.

REFERENCES

li j N. E. Abel ci a/., "TRANQUIL: A language for an array processing computer," in *Proc. AFIPS 1969 SJCC,* May 1969, pp. 57-68.

[zj R. Arnold and E. Page, "A hierarchical, restructurable multimicroprocessor architecture," in *Proc. 3rd Annu. S ymp Comput. Arch.* , Jan. 1976, pp. 40—45.

[3] G. Barnes et *al.,* "The Illiac IV computer," *IEEE Trans. Comput.,* vol. C-17, pp. 746—757, Aug. 1968.

(4] K. E. Batcher, "STARAN parallel processor system hardware," in f'roc. *AFIPS 1974 Nat. Comput. Conf.,* vol. 43, May 1974, pp. 405—410.

[5] ———, "The flip network in STARAN," in *Proc. 1976 Int. Conf. Parallel Processing,* Aug. 1976, pp. 65—71.

[6] ———, "The multidimensional access memory in STARAN," *IEEE Trans. Comput.,* vol. C-26, pp. 174—177, Feb. 1977.

[7] ———, "STARAN series E," in *Proc. 1977 Int. Conf. Parallel Processing,* Aug. 1977, pp. 140-152.

[8] ———, "MPP—A massively parallel processor," in f'ror. i979 *Int. Conf. Parallel Processing,* Aug. 1979, p. 249.

[9] J. Bogdanowicz and H. J. Siegel, "A partitionable multi-microprogrammable-microprocessor system for image processing," in I rue. *IEEE Comput. Soc. Workshop Pattern Recâg. Artificial Intell.,* Apr. 1978, pp. 141—144.

[10] W. J. Bouknight *et al.,* "The Illiac IV system," *Proc. IEEE,* vol. 60, pp. 369—388, Apr. 1972.

[11] F. Briggs, K. S. Fu, K. Hwang, and J. Patel, "PM4—A reconfigurable multimicroprocessor system for pattern recognition and image processing," in *Proc. AFIPS 1979 Nat. Comput. Conf.,* vol. 48, June 1979, pp. 255—265.

[12] Burroughs, *BSP—Burroughs Scientific Processor,* Burroughs Corp., June 1977.

[13] B. A. Crane ei at., "PEPE computer architecure," in f'roc. *COMPCON 1972, IEEE Comput. Soc. Conf.,* Sept. 1972, pp. 57—60.

[14] M. J. B. Duff, "CLIP 4: A large scale integrated circuit array parallel processor," in f'roc. *3rd Int. Joint Conf. Pattern Recog.,* 1976, pp. 728—732.

[15] A. E. Feather, L. J. Siegel, and H. J. Siegel, "Image correlation using parallel processing," in *Proc. 5th Int. Conf. Pattern Recog.* , Dec. 1980, pp. 503—507.

[16] T. Feng, "Data manipulating functions in parallel processors and their implementations," *IEEE Trarts. Comput.,* vol. C-23, pp. 309—318, Mar. 1974.

[17] P. M. Flanders *et al.,* "Efficient high speed computing with the dis- tributed array processor," in f'roc. *Symp. High Speed Comput. Algo- rithm Organization,* Apr. 1977, pp. 113—128.

[t8] M. J. Flynn, "Very high-speed computing systems," *Proc. IEEE,* vol. 54, pp. 1901–1909, Dec. 1966.

[jp] K. S. Fu, "Special computer architectures for pattern recognition and image processing—An overview," in *Proc. AFIPS 1978 Nat. Comput. Conf.,* vol. 47, June 1978, pp. 1003—1013.

[20] D. Gries, *Compiler Construction for Digital Computers.* New York: Wiley, 1971.

[21] J. Hayes, *Computer Architecture and Organization.* New York: McGraw-Hill, 1978.

[22] S. I. Kartashev and S. P. Kartashev, "A multicomputer system with dynamic architecture," *IEEE Trans. Comput.,* vol. C-28, pp. 704-720, Oct. 1979.

[231 J. Keng and K. S. Fu, "A special computer architecture for image pro- cessing," in *Proc. 1978 IEEE Comput. Soc. Conf. Pattern Recog. Image Processing,* June 1978, pp. 287-290.

[24] B. Kruse, "A parallel picture processing machine," *IEEE Trans. Comput., vol.* C-22, pp. 1075—1087, Dec. 1973.

[25] D. H. Lawrie, "Access and alignment of data in array processor," *IEEE Trans. Comput. ,* vol. C-24, pp. 1145—1155, Dec. 1975.

[26] G. J. Lipovski, "On a varistructured array of microprocessors," *IEEE Trans. Comput.,* vol. C-26, pp. 125—138, Feb. 1977.