1-1-1986

Parallel Accessible Memory

Shinji Nakamura
Dartmouth College

PARALLEL ACCESSIBLE MEMORY

Shinji Nakamura

Technical Report PCS-TR86-105

PARALLEL ACCESSIBLE MEMORY

Shinji Nakamura

Thayer School of Engineering
Dartmouth College
Hanover, NH 03755
Tel: 603/646-2343

Abstract

A new design of a memory device which allows simultaneous access to more than one location is proposed. The unique feature of this multiple accessibility of the memory is realized by applying a binomial concentrator, a type of sparse crossbar interconnection network, to content-addressable memory. The organization of the memory system and the concentration network structure as well as the network characteristics are described along with a distributive control algorithm. Applications of the memory system to parallel processing environments are also included.

This research was supported in part by a grant from the IBM Corporation, Essex Junction, Vermont.

1. Introduction

By the recent development in LSI/VLSI technology powerful computational components as well as custom designed functional blocks are available in large volume. This trend opened up a new design methodology that does not require strict limitations on the variety and quantity of hardware components. The adoption of hardware intensive parallel algorithms that can greatly improve the processing speed is possible for digital system design. Hence, the need for efficient and economical data transmission methods that support parallel processing has become increasingly important. In this paper, we propose a unique memory device with parallel access capability as a means for data transmission in parallel processing environments.

In parallel processing applications, depending on the communication requirements, there have been several different types of interconnection networks proposed in the literature which are able to establish routes between various combinations of processing elements; for example, permutation networks [1]-[3], for all possible one-to-one connections between source units to destination units, omega networks [4] for restricted permutations, SIMD interconnection networks [5] for Single-Instruction Multiple-Data Stream machines, and Banyan networks [6] for partition or reorganization of multiple computing module systems. In general, those approaches are made by directly applying the interconnection networks to the system components, such as processing units, memory modules, and I/O devices, so that the interconnection network used in an application is very much dependent on the application and the control algorithm for the network and is usually complicated. In contrast to these approaches, the proposed method resolves the data transmission-related problems by an indirect application of an interconnection network to the processing units through a memory device. It is interesting to note here that

the conventional memory is inherently sequential; the address has to be unique to specify one particular word among many, and this limits the capability of accessing more than one word at a time. The only parallelism that can be introduced to the conventional memory would be to have a longer word length. Therefore, for our application in memory accessing we adopt content-addressing [7] instead of the conventional addressing. By applying a concentration network to the parallel readout lines of the memory, multiple word information can be routed to multiple processing units in parallel.

In this paper, we will discuss the memory and the overall system organization in Section 2, the concentration network and its distributive control in Section 3, remarks about extendability and fault tolerance in Section 4, and in Section 5 we will include applications of the memory device to parallel processings.

2. Content Addressable Memory and System Organization

The entire memory system is organized by content addressable memory, referred to as CAM, and a concentrator. Although the write operation of the memory does not have special features, the read operation can be used simultaneously for many words. The content addressing selects all the words that satisfy a read out condition. The selected words are transferred to the concentrator through wires that make distinct connections between CAM words and the inputs to the concentrator. The number of output lines of the concentrator, which are the read out lines of the memory system, is substantially smaller than the number of CAM words, so that if the number of selected words is more than the capacity of the concentrator, the concentrator first reduces the number of selected words and decides which word is going to be routed to which read out line of the system. A block diagram of the memory system is shown in Figure 1.

The main part of the CAM has the similar configuration of the conventional memory. It is organized as an array of bit registers, which are grouped into a set of cells, or words, of appropriate length. However the accessing method for a word in CAM is completely different. For CAM, a piece of information, called tag, and the partial contents of every memory cell are required for memory access. Figure 1 shows a block diagram of a CAM device. There are two special registers, reference register and mask register, which have the same length of memory word. When the memory is read, the reference register is masked by the mask register and produces a tag. All the memory words are also masked by the mask register and they are compared with the tag. The masking and comparison operations are performed on all the words simultaneously. If a word has the same contents of the tag it is said that the word has a match.

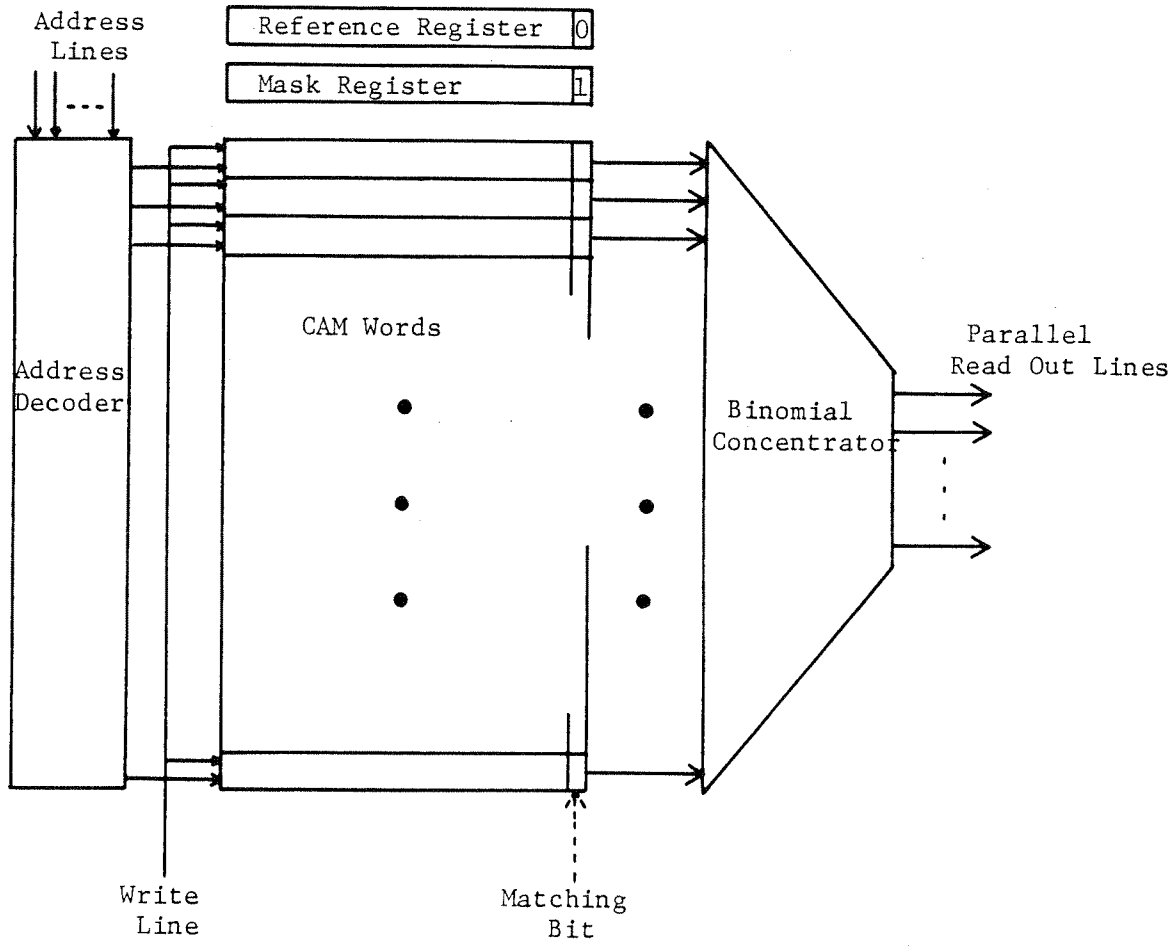

Figure 1. A block diagram of CAM and the concentrator.

Because this access method is very similar to the table lookup search by the reference word, the operation of finding matches is called a search. Every CAM word is connected to the concentrator, so that all the matched words are transmitted to the concentrator. But depending on the number of matched words and the circumstance, some of the matched words may not be routed to any of the output lines of the concentrator. It also should be pointed out that from the property of the proposing concentrator the assignment of a particular output line of the concentrator to a particular matched word usually cannot be specified. This aspect of the concentrator is discussed in more detail in the next section. The matched word which was not routed to the output end of the concentrator is called a hidden match. For those hidden matches, a matching bit is included to each CAM word and the reference and masked registers. The matching bits in the reference register and mask register are always 0 and 1, respectively. The matching bits in CAM word are first cleared to 0 before a search is made. When a match is made and the matched word gets an output path through the concentrator, the match bit of the word is set to 1. If there is at least one hidden match and when the second search is tried, those words which have been routed out will not make matches because their matching bits are 1 and the matching bit in the reference word is 0.

The address lines, address decoder and write line are only used for the write operation to a CAM word. The write operation is made for one word at a time by specifying a unique address for the word and there is no difference from the write operation of the conventional memory.

As a whole system, the CAM with the output concentrator has bit-serial word-parallel read operation by content addressing, and bit-serial word-serial write operation by conventional addressing.

3. Concentrator and the Control Algorithm

As it is briefly discussed in the previous section, the concentration network resolves the multiple matches of CAM by routing them in parallel to the output lines. If there are more multiple matches than the capacity of the network some of them have to be blocked. The network we propose to the memory system which will perform the above mentioned task is a special type of sparse crossbar concentrator called a binomial concentrator [8]. Since the binomial concentrator has a simple structure and a high throughput capacity with a distributed control, it was found to be most appropriate for this application.

In this section the characteristics and a control algorithm for the binary concentrator are described.

A sparse crossbar (n,m,c)-concentrator is a bipartite graph N with the bipartition (I,O), where $|I| = n$, $|O| = m$, $n > m$, and I and O are called, respectively, the input and output sets. N is said to have the capacity c, $c \leq m$, if for any choice of k inputs, $k \leq c$, there exists a set of k edges that connect each input to some distinct output [9]. When N is expressed in a crosspoint diagram, as in Figure 2, I and O are defined by columns and rows,

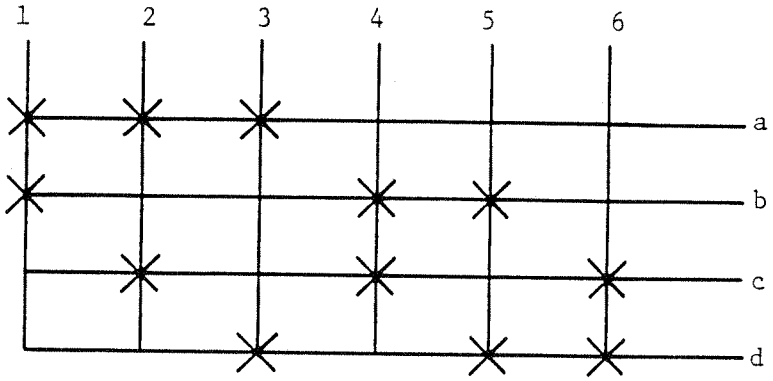

Figure 2. A (6,4,4,)-concentrator.

and an edge between i and o, $i \varepsilon I$ and $o \varepsilon O$, is expressed by the presence of a crosspoint "×" on the cross between the i column and the o row. The fact that the crosspoint diagram can be viewed as a crossbar network with some crosspoints removed and the condition n > m has resulted in the name "sparse crossbar concentrator."

In the above example, I = {1,2,3,4,5,6} and O = {a,b,c,d}. The capacity c of the network is 4, which is verified by examining all $\binom{6}{1} = 6$, $\binom{6}{2} = 15$, $\binom{6}{3} = 20$, $\binom{6}{4} = 15$ possible input sets of size 1,2,3, and 4. Every input in each input set is incidental to some distinct output through a crosspoint. N is conventionally expressed by the three numbers $(|I|,|O|,c)$. When $|O| = c$, N is said to have full capacity.

A binomial $\binom{m}{v}$-network [9] is a sparse crossbar $(\binom{m}{v},m,v+2)$ concentrator, where each of the $\binom{m}{v}$ inputs has crosspoints to a unique choice of v of the m outputs. The example shown in Figure 2 exhausts all possible $\binom{4}{2}$ combinations of crosspoint placement so that it is a $\binom{4}{2}$-network, which is (6,4,4) full capacity concentrator. The capacity of an $\binom{m}{v}$-network was first discussed in [8]. A design for a full capacity concentrator requiring a minimum number of crosspoints was given in [10], and the binomial $\binom{m}{v}$-network was shown to be a minimum [10].

From the definition of the concentrator it is clear that one-to-one correspondence between input and output is not arbitrarily specified. For instance, in Figure 2 input 1 cannot be routed to output c or d. Hence, for those applications where specific output destination is required, another permuting network should be necessary at the output end of the concentrator. However, the size of the permuter depends on the number of output lines of the concentrator, which is small, so that this approach by the binomial concentrator with a small permuter is still more advantageous in many cases than

other types of powerful but more complicated networks.

The capacity c defined above is the number that up to c any inputs can be simultaneously routed to distinct outputs. And binomial concentrators yield certain capacities with the minimum number of crosspoints. But in real application the average capacity gives a better measure for the network throughput. The average capacity of $\binom{m}{2}$-networks was studied in [11] which showed very high value, close to the number of output lines m. Since a concentrator is a bipartite graph, the problem of finding disjoint paths between subsets of inputs and outputs of a concentrator is identical to solving the maximum matching problem in the corresponding bipartite graph. Hence, any algorithm that solves the maximum matching problem can also control the concentrator and realize the throughput as to the capacity and average capacity. There are known good algorithms [12] for the maximum matching problem, but because they are designed for general bipartite graphs with serial processing, the application of these algorithms to a concentrator does not result in high efficiency in speed and area. Therefore, instead of these general algorithms we apply a straightforward distributive algorithm [13], which will yield the maximum throughput for the full capacity concentrators to the binomial concentrator, although the actual throughput becomes slightly less than the calculated capacities.

The control algorithm is based on the precedence rules for designating left and up. If we orient the crosspoint diagram as shown in Figure 2 (i.e., inputs from the top and outputs to the right) and assign a higher control priority to the left and up signals and propagate control signals through the crosspoints from top to bottom and left to right, the condition for the "on" state of a crosspoint can be stated as: If a connection request signal arrives from the top and there is no busy signal from the left, then the next state of

the crosspoint is "on." This rough description of the control mechanism can be specified more precisely by applying two-dimensional iterative array logic [14] where each crosspoint cell communicates with upper, lower, left, and right neighboring cells. Figure 3 is a model of a crosspoint cell where x and y are inputs from the cells left and upper neighbors, and S is the "on"/"off" state of the crosspoint.

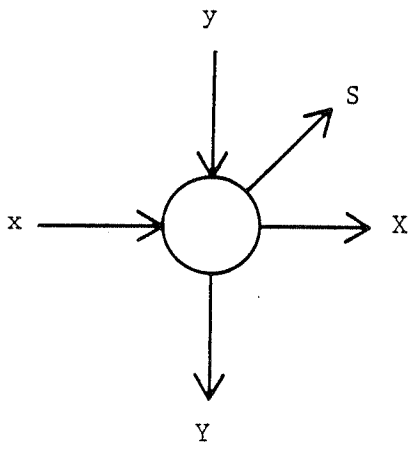

Figure 3. A crosspoint cell for the control algorithm.

The transition rules of the crosspoint are defined by explicitly stating the output values X, Y, and S in terms of x and y input values, such as defined in Figure 4.

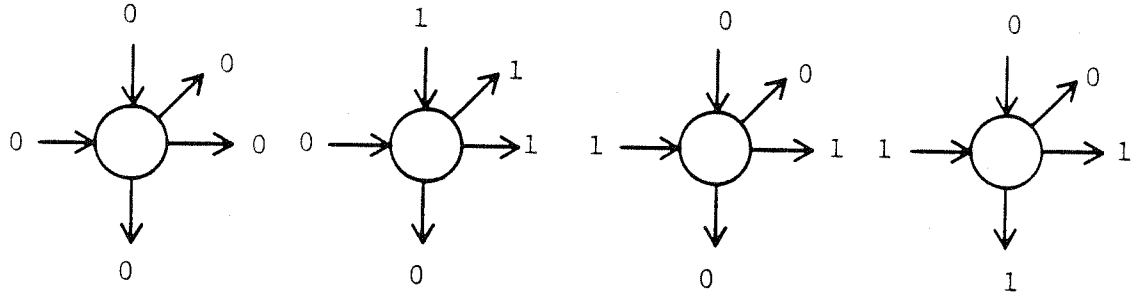

Figure 4. The transition diagram for a crosspoint cell.

From these transition rules X, Y, and S are expressed in Boolean expressions:

$$X = x+y, \qquad Y = xy, \qquad S = xy.$$

It is shown that with the above logic at each crosspoint of a $\binom{m}{v}$-network the actual capacity realized by the control algorithm becomes v+1, [13]. In Table 1 the actual average capacities realized by the control algorithm are computed by computer simulations with 3000 sample points for each calculation. The results from $\binom{m}{2}$-network cases were compared with the true values computed in [11]. In the table, parenthesized values show the true average capacity values. The comparisons on the table show the inefficiency in throughput introduced by the control algorithms is very small. With the straightforward distributive control algorithm the binomial concentrator is an efficient practical approach for the parallel accessible memory system.

Number of Input	$\binom{12}{2}$	$\binom{12}{3}$	$\binom{13}{2}$	$\binom{13}{3}$	$\binom{14}{2}$	$\binom{14}{3}$	$\binom{15}{2}$	$\binom{15}{3}$
1	(1.000) 1.000	1.000	(1.000) 1.000	1.000	(1.000) 1.000	1.000	(1.000) 1.000	1.000
2	(2.000) 2.000	2.000	(2.000) 2.000	2.000	(2.000) 2.000	2.000	(2.000) 2.000	2.000
3	(3.000) 3.000	3.000	(3.000) 3.000	3.000	(3.000) 3.000	3.000	(3.000) 3.000	3.000
4	(4.000) 3.992	4.000	(4.000) 3.994	4.000	(4.000) 3.995	4.000	(4.000) 3.998	4.000
5	(4.999) 4.975	5.000	(4.999) 4.976	4.999	(4.999) 4.982	5.000	(4.999) 4.984	5.000
6	(5.997) 5.915	5.994	(5.999) 5.934	5.997	(5.999) 5.948	5.996	(5.999) 5.949	5.999
7	(6.987) 6.823	6.974	(6.992) 6.848	6.986	(6.995) 6.880	6.992	(6.999) 6.886	6.995
8	(7.954) 7.640	7.928	(7.973) 7.716	7.953	(7.983) 7.779	7.972	(7.989) 7.784	7.974
9	(8.367) 8.427	8.827	(8.921) 8.516	8.875	(8.951) 8.601	8.912	(8.969) 8.649	8.941
10	(9.671) 9.044	9.609	(9.800) 9.233	9.726	(9.876) 9.375	9.819	(9.921) 9.441	9.875
11	(10.307) 9.631	10.222	(10.562) 9.872	10.485	(10.721) 10.078	10.632	(10.820) 10.178	10.750
12	(10.743) 10.054	10.726	(11.158) 10.450	11.081	(11.442) 10.659	11.355	(11.630) 10.814	11.544
13			(11.578) 10.857	11.575	(12.005) 11.213	11.977	(12.314) 11.433	12.273
14					(12.414) 11.639	12.470	(12.350) 11.944	12.825
15							(13.250) 12.405	13.301

Table 1. Actual average capacity of $\binom{m}{v}$-network using the control algorithm. Values in parentheses are true average capacities.

4. Extendability and Fault Tolerance

In a CAM word when accessing words are selected a comparison with the tag is made by the comparator for the word. Because the comparator requires information from many bits of the word, words in the memory cannot be sliced into bit planes. Therefore, the memory has to be organized in bit-serial fashion, which prevents extension of word length, and the only possible memory extension is by increasing memory words. The actual implementation of the memory system will include the entire organization of reference and mask registers, CAM words, and concentrator, shown in Figure 1 in a single chip, so that the simple extension by adding another chip will double the number of output lines. A reduction of output lines in an extended memory can be made by cascading another binary concentrator to the output lines of the extended memory. Figure 5 shows such an extension. For this type of extension the control algorithms for the cascading as well as cascaded concentrator are almost the same as the one described in the previous sections. The only difference is that the blocking information is generated not only in the first concentrator but also in the second stage so that the first stage concentrator has to be modified to accept this information and pass it to CAM words. There is also a slight difference in the average capacity. If we look at parallel outputs from a single first stage memory block, the average capacity is decreased by the effect of the second stage concentrator. Suppose an expanded memory system was made by cascading $\binom{12}{2}$-networks and there are 10 parallel outputs made from a single first stage memory block. From Table 1 the average capacity from the first stage is 9.004 but the average capacity from the second stage will reduce to 8.427 since the average capacity for 9 inputs at the second stage is 8.427. The memory expansion of this type by more than one cascading is possible and the reduction rate of the average capacity is relatively small even for multiple cascading.

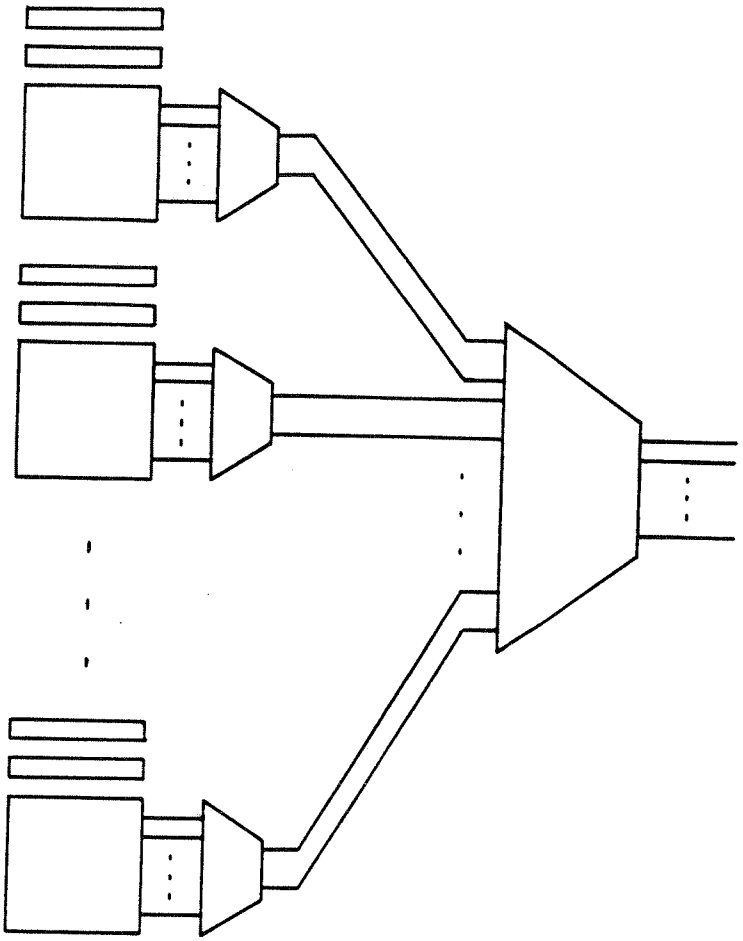

Figure 5. A memory extension by cascading concentrators.

Another interesting feature of the memory system is fault tolerance. From the property of the binomial concentrator all CAM words are connected more than one output lines of the concentrator through crosspoints. Therefore, even if there is a faulty device attached to an output line, the system still can function correctly by blocking the output line. The control algorithm is designed to accept a busy signal from the left neighbor and the blocking operation can be made by sending a busy signal to the leftmost crosspoint on

the line. With an $\binom{m}{2}$-network there is exactly one CAM word which is connected to a specific pair of output lines. Hence, for a $\binom{m}{2}$-network two faulty devices make only one word inaccessible to a correct device. In general, for a $\binom{m}{v}$-network, v faults make one word inaccessible and k faults, $v \leq k \leq m$, make $\binom{k}{v}$ words inaccessible to a correct device.

5. Applications

In this section, we discuss some of the data analysis/recognition-type applications of the parallel accessible memory. There are two different ways the memory system is used. The first method uses the CAM as a buffer of the input data, and the reference register to store the condition with which a set of input data that meets the condition is selected. This is applicable to image processing, sequence detector, etc., where a particular spot of specific context is searched among the input data. In the second method, the role of the CAM and the reference register is interchanged. Before the CAM is searched, some key information is extracted from the input data and stored into the reference register. The CAM contains the possible variety of such keys. This second method can be used for context analysis of input data where different context requires different processing. We discuss this second case in more detail.

Figure 6 shows the entire configuration of the application. At each output line of the memory system, a processing unit, referred to as PU, is attached. All the PUs have individual memory and their hardware and software are identical. Depending on the specific application, a PU can be an ordinary microprocessor or a specially designed hardware controlled by micro code sequence. The entire system is controlled by a supervising processor, SP. The main task of the SP is input handling and system management. The input data stream is accepted by the SP. While the SP preprocesses the input data to form a key for the reference word and a mask for the mask register, the input data is broadcasted to all the PUs and stored in the individual memory. Therefore, when the CAM is ready to be searched, every PU has the same setup with a copy of input data. A CAM word contains a key and a program location for PU. When a match is made, the PU connected to the matched word through the concentrator

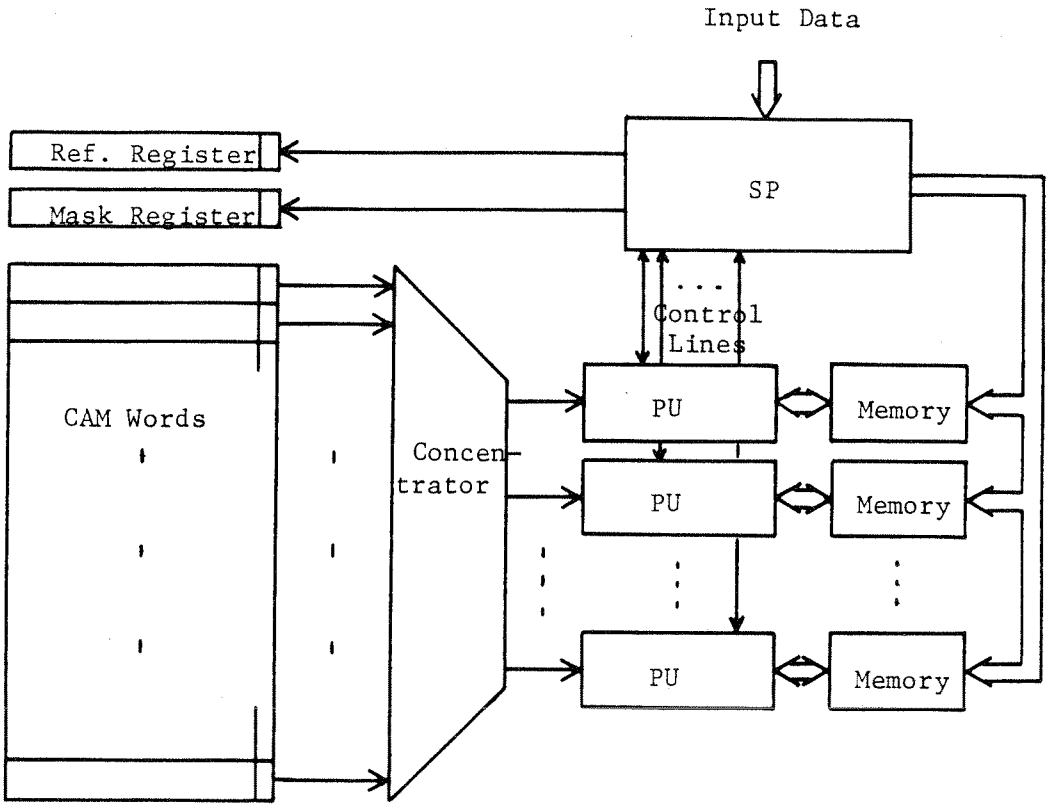

Figure 6. An application example of the parallel accessible memory.

will be activated by starting its execution from the location specified in the matched word. If there are m multiple matches, m different analysis for the same input data is performed in parallel by m active PUs. After processing their own copy of input data, the active PUs will report their results to the SP. If all the matches were not resolved, a flag will be set "on." The SP senses this situation through a control line from CAM, and reinitiates a search with the same reference and mask word. This time, match bits are not cleared before the access so that the new words are going to be selected.

Depending on the specific applications, many variations of the above-described architecture are possible. For instance, if the main process requires very simple operations such as total summation of the matched words, all the PUs can be replaced by a parallel adder; or if the process does not change the input data, each PU does not have to have its own memory, instead a single memory area is shared by all the PUs and SP.

As a more concrete example of the system, we consider recognition of hand-written alphabet characters. The underlying idea of this application is a two-step recognition process. In the first step, a vague prediction of possible results is made, and in the second step each of the possibilities are checked in parallel and one of them is going to be the correctly recognized character. The first recognition step is made in SP. When data for a new character is read into the input buffer, the SP makes a brief profile of the input character from statistical information, such as the ratios of the number of black and white pixles in different regions. From the profile, predictions about the possible character can be made. A CAM word corresponds to a character with a possible profile for the character. In addition to a profile, a word contains a program location where the second step recognition process of the character begins. With a profile in the reference word a search on the CAM

will find matches. Each match will initiate an independent recognition process on a PU with a prediction of a certain character. Since each segment of the PU program can be designed to recognize one particular character, the algorithms should be faster and simpler than the one that deals with general cases.

6. Conclusion

In this paper, we have proposed a new parallel accessible memory system. The content addressing scheme and a binomial concentrator were employed in the memory system to provide parallel read out capability. The throughput capacity of the parallel read operation was discussed with a control algorithm for the concentrator, and the memory system was found to have high throughput efficiency by practical distributive control. Although the memory was organized in bit-serial format, memory extension is possible by cascading output concentrators. A fault tolerant feature was also discussed and the overall feasibility of the memory system to actual parallel processing environments was shown by application examples.

References

1. V. Benes, "Mathematical Theory of Connecting Networks and Telephone Traffic," Academic Press, New York, 1965.
2. C. Clos, "A Study of Non-Blocking Switching Networks," Bell System Tech. J., Vol. 32, No. 2, Mar. 1953, pp. 406-424.
3. P. Cantor, "On Construction of Nonblocking Switching Networks," Proc. of Symposium on Computer-Communications Networks and Teletraffic, Polytechnic Institute of Brooklyn, 1972.
4. D.H. Lawrie, "Access and Alignment of Data in an Array Processor," IEEE Trans. Comput., Vol. C-24, No. 12, Dec. 1975, pp. 1145-1155.
5. H. Siegel, "Single Instruction Stream Multiple Data Stream Machine Interconnection Network Design," Proc. 1976 Intl. Conf. on Parallel Processing, 1976, pp. 273-282.
6. L. Goke and G. Lipovski, "Banyan Networks for Partitioning Multiprocessor Systems," Proc. 1st Annual Comput. Architecture Conf., 1973, pp. 21-28.
7. T. Kohonen, "Content-Addressable Memories," Springer-Verlag, New York, 1980.
8. G.M. Masson, "Binomial Switching Networks for Concentration and Distribution," IEEE Trans. Comm., Vol. COM-25, No. 9, 1977, pp. 873-883.
9. G.M. Masson, G.C. Gingher, and S. Nakamura, "A Sampler of Circuit Switching Networks," IEEE Computer, Vol. 12, No. 6, 1979, pp. 32-48.
10. S. Nakamura and G.M. Masson, "Lower Bounds on Crosspoints in Concentrators," IEEE Trans. Comp., Vol. C-31, No. 12, pp. 1173-1179.
11. G.M. Masson and S.B. Morris, "Expected Capacity of $\binom{m}{2}$-Networks," IEEE Trans. Comp., Vol. C-32, No. 7, July 1983.

12. J.A. Bondy and U.S.R. Murty, "Graph Theory with Applications," North Holland, 1979.

13. S. Nakamura, "Control Algorithms for Sparse Crossbar Concentrators," Proc. 18th Annual Conf. on Information Sciences and Systems, 1984.

14. Z. Kohavi, "Switching and Finite Automata Theory," 2nd ed., McGraw Hill, 1978.