INTERCOMMUNICATING CELLS, BASIS FOR A DISTRIBUTED LOGIC COMPUTER

C. Y. Lee
Bell Telephone Laboratories, Inc.
Holmdel, New Jersey

The purpose of this paper is to describe an information storage and retrieval system in which logic is distributed throughout the system. The system is made up of cells. Each cell is a small finite state machine which can communicate with its neighboring cells. Each cell is also capable of storing a symbol.

There are several differences between this cell memory and a conventional system. With logic distributed throughout the cell memory, there is no need for counters or addressing circuitry in the system. The flow of information in the cell memory is to a large extent guided by the intercommunicating cells themselves. Furthermore, because retrieval no longer involves scanning, it becomes possible to retrieve symbols from the cell memory at a rate independent of the size of the memory.

Information to be stored and processed is normally presented to the cells in the form of strings of symbols. Each string consists of a name and an arbitrary number of parameters. When a name string is given as its input, the cell memory is expected to give as its output all of the parameter strings associated with the name string. This is called direct retrieval. On the other hand, given a parameter string, the cell network is also expected to give as its output the name string associated with that parameter string. This is called cross-retrieval.

The principal aim of our design is a cell memory which satisfies the following criteria:

1. The cells are logically indistinguishable from each other.
2. The amount of time required for direct retrieval is independent of the size of the cell memory.
3. The amount of time required for cross-retrieval is independent of the size of the cell memory.
4. There is a simple uniform procedure for enlarging or reducing the size of the cell memory.

Aim and Motivation

We are primarily concerned here with the design of the memory system of a computer in which memory and logic are closely interwoven. The motivation stems from our contention that in the present generation of machines the scheme of locating a quantity of information by its "address" is fundamentally a weak one, and furthermore, the constraint that a memory "word" may communicate only with the central processor (in most cases the accumulator) has no intrinsic appeal. This motivation led us to the design of a cell memory compatible with these contentions.

The association of an address with a quantity of information is very much the result of the type of computer organization we now have. Writing a program in machine language, one rarely deals with the quantities of information themselves. A programmer normally must know where these quantities of information are located. He then manipulates their addresses according to the problem at hand. An address in this way often assumes the role of the name of a piece of information.

There are two ways to look at this situation. Because there is usually an ordering relationship among addresses, referring to contents by addresses has its merits provided a programmer has sufficient foresight at the time the contents were stored. On the other hand, a location other than being its address can have but the most superficial relation to the information which happens to be stored there. The assignment of a location to a quantity of information is therefore necessarily artificial. In many applications the introduction of an address as an additional characteristic of a quantity of information serves only to compound the complexity of the issue. In any event the assignment of addresses is a local problem, and as such should not occupy people's time and may even be a waste of machine's time.

A macroscopic approach to information storage and retrieval is to distinguish information by its attributes. A local property, "350 Fifth Avenue," means little to most people. The attribute, "the tallest building in the world," does. The macroscopic approach requires only that we be able to discern facts by contents. Whatever means are needed for addressing, for counting, for scanning and the like are not essential and should be left to local considerations.

Doing away with addressing, counting, and scanning means a different approach to machine organization. The underlying new concept is however simple and direct: The work of information storage and retrieval should not be assumed by a central processor, but should be shared by the entire cell memory. The physical implementation of this concept is intercommunicating cells.

Although one of the principal aims of an intercommunicating cell organization is to make the rate of retrieval independent of the amount of information stored in the cell memory, a number of other engineering criteria are no less important. Uniformity of cell design makes mass production possible. Ease of attaching or detaching cells from the cell memory simplifies the growth problem. Also, facilities for simultaneous matching of symbols make complex preprocessing such as sorting unnecessary.

A few intuitive remarks on cell memory retrieval may be appropriate here. Information stored in the cell memory are in the form of strings of symbols. Normally, such a string is made up of a name and the parameters which describe its attributes. Each cell is capable of storing one symbol. A string of information is therefore stored in a corresponding string of cells. In the cell memory each cell is given enough logic circuitry so that it can give us a yes or no answer to a simple question we ask. If we think of the symbol, say s, contained in a cell as its name, then the question we ask is merely whether the cell's name is s or is not s.

In retrieval, for example, we may wish to find all of the parameters (attributes) of a strategy whose name is XYZ. As a first step we would simultaneously ask each cell whether its name is X. If a cell gives us an answer no, then we are no longer interested in that cell. If a cell gives us an answer yes, however, we know it may lead us to the name of the strategy we are looking for. Therefore, we also want each cell to have enough logic circuitry so that it can signal its neighboring cell to be ready to respond to the next question. We then simultaneously ask each of these neighboring cells whether its name is Y. Those cells whose names are Y in turn signal their neighboring cells to be ready to respond to the final question: whether a cell's name is Z. The cell which finally responds is now ready to signal its nearest neighbor to begin giving out parameters of the strategy.

The process of cell memory retrieval provides a particularly good example of letting the cells themselves guide the flow of information. By a progressive sequence of questions, we home in on the information we are looking for in the cell memory, although we have no idea just where the information itself is physically stored. Because generally most cells contain information which is of no use to us, the number of cells which give yes answers at any moment is quite small. We may, if we wish, think of the retrieving process therefore as a process of getting rid of useless information rather than a searching process for the useful information.

Cell Configuration

Each cell in the cell memory is made up of components called cell elements. Each cell element is a bistable device such as a relay or a flip-flop. The cell elements are divided into two kinds: the cell state elements and the cell symbol elements. In the design of the cell memory to be described here, each cell will have a single cell state element so that each cell has two logical states. A cell may either be in an active state or in a quiescent state. There may be any number of cell symbol elements, depending upon the size of the symbol alphabet.

The over-all structure of a cell memory is shown in Figure 1. Each cell in the cell memory, say cell i, is controlled by four types of control leads: the input signal lead (IS lead), the output signal lead (OS lead), the match signal lead (MS lead), and the propagation lead (P lead). The input signal lead is active for the duration of the input process. The input symbol itself is carried on a separate set of input leads. When a cell is in an active state, and the input signal lead is activated, whatever symbol is carried on the input leads is then stored in that cell.

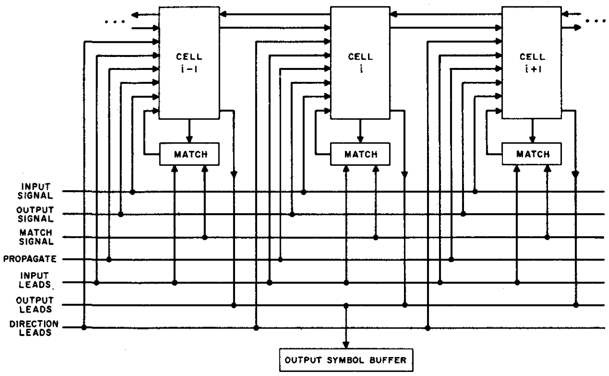

Figure 1. Overall diagram of the cell memory.

The output signal lead controls the flow of output from the cell memory. Each symbol read out from a cell is carried by a separate set of output leads, and is also stored in a buffer called the Output Symbol Buffer. When a cell is in an active state, a pulse on the output signal lead causes that cell to read out its contents to the set of output leads.

An important function performed by the cell memory is the operation of simultaneous matching of the contents of each of the cells with some fixed contents. This operation is controlled by the match signal lead. During matching, the contents of each cell, say cell i, is compared with the contents carried on the input leads. If the comparison is successful, an internal signal, m_i, is generated in cell i. The signal, m_i, is transmitted to one of the neighboring cells of cell i, causing that cell to become active.

The propagation lead controls the propagation of activity in a cell memory. When a cell is in an active state, a pulse on the propagation lead causes it to pass its activity to one of its neighboring cells. The direction of propagation is controlled by two separate direction leads: R and L.

The circuits of a cell employing flip-flops and gates are shown in Figure 2.

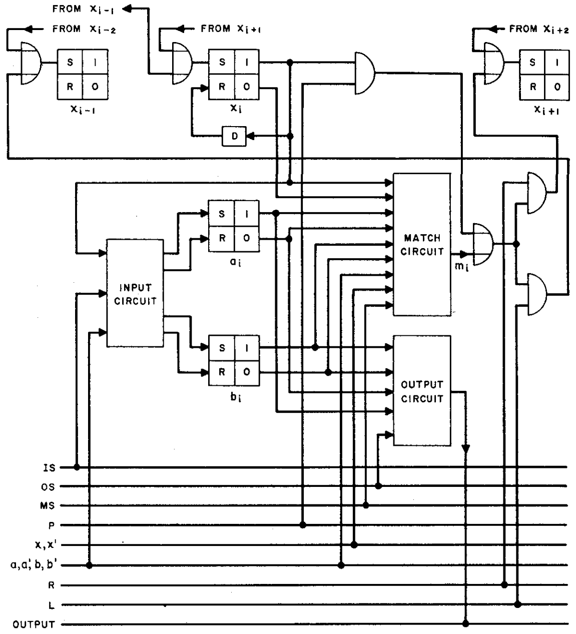

Figure 2. More detailed circuit of cell i.

An Example of Cross-Retrieval

Let there be three separate strings of information stored in the cell memory. Let these strings have the following names and parameters:

Name	Parameter
B	XY
AB	XW
AC	U

The strings are stored in the cell memory in the form of a single composite string. There must be some way, therefore, by which the name and the parameter strings can be told apart and also a means for distinguishing among the three strings of information themselves. To do this we introduce two tag symbols, α and β. Every name string is preceded by a tag of α, and every parameter string is preceded by a name of β. The string stored in the cell memory therefore has the form:

α B β X Y α A B β X W α A C β U α ...

We have found it convenient to use the diagram

$$\boxed{p \mid q}$$

to represent a cell. In this diagram, p stands for the symbol stored in the cell, and q for the state of the cell. Also, q is 1 if the cell is quiescent, and is 2 if the cell is active. Using such diagrams, Figure 3 shows the manner in which the composite string is stored in the cell memory.

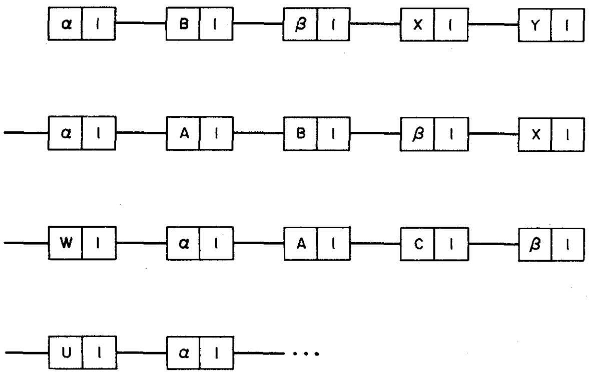

Figure 3. Cell memory configuration at the start of retrieval.

Let us suppose that we wish to retrieve the name of a string whose parameter is XW. We call such a process in which a parameter string input causes a name string output the process of cross-retrieval. The process of retrieving a parameter knowing its name is called direct retrieval.

Initially, we want all of the cells to match their individual contents against the fixed information

$$\boxed{\beta \mid 1}$$

Furthermore, we want every cell whose contents happen to be $\boxed{\beta \mid 1}$ to send a signal to the neighboring cell on its right. We then have the situation shown in Figure 4, where each arrow indicates that a signal is being transmitted by a cell whose contents are $\boxed{\beta \mid 1}$.

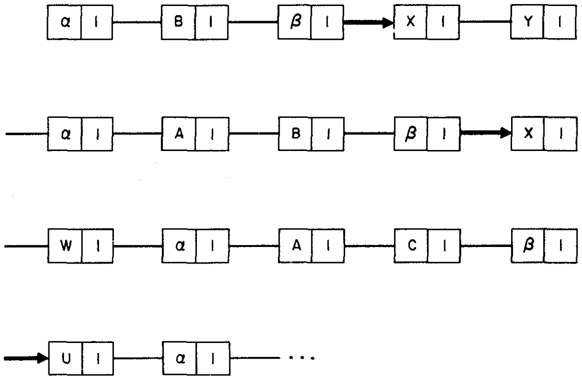

Figure 4. Signals being transmitted by $\boxed{\beta \mid 1}$ after matching.

Now every cell which receives a signal from its neighboring cell, whether from the left or from the right, will change from a quiescent state to an active state. Also, the signals transmitted by the cells to their neighbors should be thought of as pulses so that they disappear after they have caused the neighboring cells to become active. The next stable situation is shown in Figure 5; each of the active cells is represented by double boundary lines.

During the next match cycle, we want all of the cells to match their individual contents against the fixed contents:

$$\boxed{X \mid 2}$$

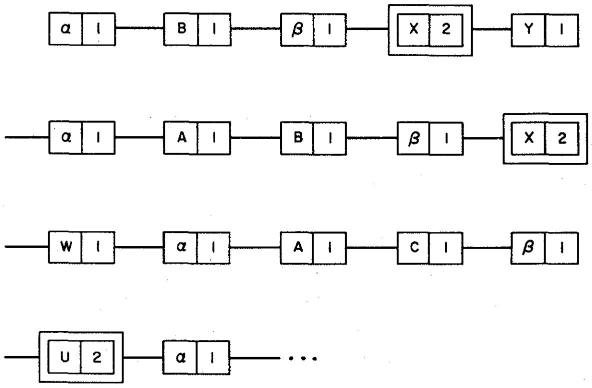

Figure 5. Some of the cells become active after receiving pulses from the neighboring cells.

As before, every cell whose contents are $\boxed{X \mid 2}$ sends a signal to its right neighboring cell, causing that cell to become active. At this point, each previously active cell is made to restore itself to the quiescent state. The stable situation is illustrated in Figure 6.

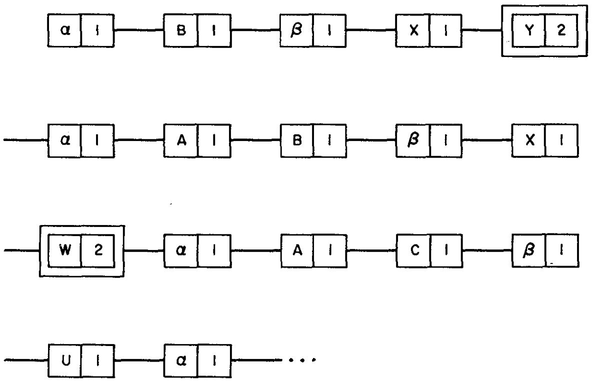

Figure 6. Previously active cells restore themselves to quiescent state as new cells become active.

During the following match cycle, the cells are made to match their individual contents against the fixed contents

$$\boxed{W \mid 2}$$

In this example, there is now only one cell whose contents are $\boxed{W \mid 2}$. That cell first signals the cell on its right, causing that neighboring cell to become active, and then restores itself to the quiescent state.

During the next match cycle, all of the cells are made to match against

$$\boxed{\alpha \mid 2}$$

The presence of the symbol α shows that the matching process is at an end, and that the output part of the retrieval process is to begin. The cell whose contents are $\boxed{A \mid 2}$ is the only cell which is active at the moment.

During the output phase, a number of actions take place. First of all, two successive propagate-left signals are sent to the cell memory. The result is a transfer of activity from the cell whose contents are $\boxed{A \mid 2}$ to the cell whose contents are $\boxed{W \mid 2}$, as shown in Figure 7. An output signal is now supplied to all of the cells. The cell which is active then reads out its symbol to the output leads. That symbol is now compared with the fixed symbol α in an external match circuit associated with the control leads. If there is no match, a propagate-left signal is sent to every cell and the external comparison process is repeated. This process eventually terminates when the cell $\boxed{\alpha \mid 2}$ is reached (Figure 8). The purpose of this phase of the output process is strictly to locate the beginning of the information string which is being retrieved.

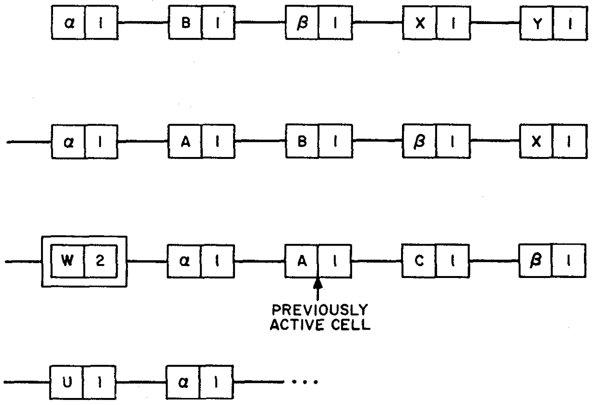

Figure 7. Transfer of activity from cell $\boxed{A \mid 2}$ to cell $\boxed{W \mid 2}$

The actual read out process begins with a propagate-right signal. The cell $\boxed{A \mid 2}$ which contains the first letter of the name string AB is activated. The symbol A is now read out and matched externally with the symbol β. The read out process continues until β is reached and is then terminated.

Figure 9 gives a general picture of the read out part of the cross-retrieval process.

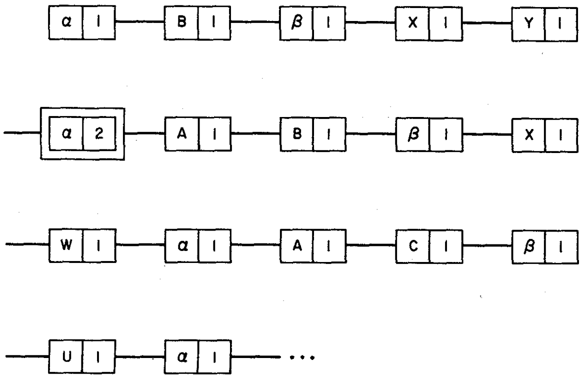

Figure 8. Reaching the initial symbol in the string to be retrieved.

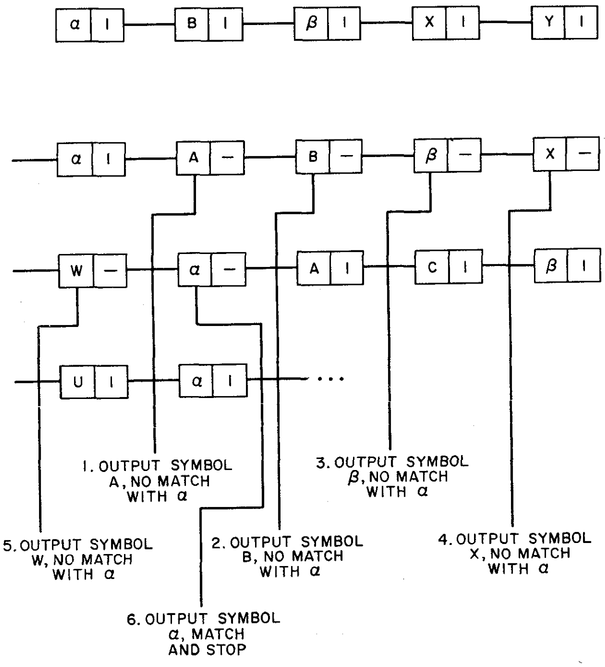

Figure 9. A general picture of the read-out part of the cross-retrieval process.

Storage and Cross-Retrieval

The storage and the retrieval of symbols in the cell memory are both accomplished by letting the cells pass their activities to their neighboring cells and, in this way, guide the flow of information in the cell memory. The process of storing symbols in the cell memory provides a good illustration of the dependence of the cell memory upon the propagation of activity among the cells.

Prior to storing the first symbol, the first cell in the cell memory is made active. When the first symbol appears on the set of input leads, the first cell, being active, becomes the only cell prepared to receive that symbol. The first cell, like all of the cells, plays a dual role however. After taking in the symbol, it also passes its activity to the right neighboring cell. The neighboring cell then becomes the only active cell in the cell memory, and hence becomes the only cell prepared to receive the next symbol when it appears on the input leads.

When we examine the many kinds of information strings which make retrieval difficult, we find that a string is much more likely to have several parameters rather than a single parameter. For such strings the tag system which we have used in the example in the last section is inadequate. If a string consists of a name N and k-1 parameters P_1, P_2, ..., P_{k-1}, we will now assign to it a set of k+2 tags: α_0, α_1, ..., α_k, and β. The string will be stored in the cell memory in the following form:

$$\ldots \alpha_0\ \alpha_1\ N \beta \alpha_2\ P_1\ \beta \alpha_3\ P_2 \beta \ldots$$

$$\alpha_k\ P_{k-1}\ \beta \alpha_0 \ldots$$

The symbol α_0 indicates the beginning of a string, and the symbol β indicates the end of a component (that is, a name or a parameter). The symbols α_1, α_2, ..., α_k are the tags associated with the components N, P_1, ..., P_{k-1}. Furthermore, it should be noted that a tag is associated always with a given attribute. For example, α_1 is the name tag and should be used as a name tag for all information strings.

Consider now the cross-retrieval problem where the cell memory is given as its input a component string together with its tags:

$$\alpha_j\ P_{j-1}\ \beta$$

The cell memory, for the purpose of cross-retrieval, must give as its output the entire string:

$$\alpha_1\ N \beta \alpha_2 P_1\ \beta \ldots \alpha_k\ P_{k-1}\ \beta$$

In describing a procedure for cross-retrieval, we shall assume for the purpose of this presentation that every component stored in the cell memory is unique. This means that if an input string $\alpha_j\ P_{j-1}\ \beta$ is presented to the cell memory, we can be sure that either (1) there is no string in the cell memory which has $\alpha_j\ P_{j-1}\ \beta$ as one of its components, or (2) there is exactly one string in the cell memory which has $\alpha_j\ P_{j-1}\ \beta$ as one of its components. Under this assumption therefore, no two strings could compete with each other during retrieval.

The basic cross-retrieval procedure is the following. The string $\alpha_j\ P_{j-1}\ \beta$ is first matched with all of the strings stored in the cell memory. When a match has occurred, the cell in which the symbol α_{j+1} is stored (α_{j+1} being, in this case, the symbol immediately following $\alpha_j\ P_{j-1}\ \beta$ in the cell memory) would be activated. Because $\alpha_j\ P_{j-1}\ \beta$ is unique in the cell memory, the cell in which the symbol α_{j+1} is stored becomes the only active cell in the cell memory. This activity is now propagated towards the left until the symbol α_0, which is the beginning of the string, is reached. Symbols are then retrieved from the cell memory to the right until finally the symbol α_0, which is the beginning of the next string, is reached.

Outlook

We wanted to present here the basic ideas of a distributed logic system without going into many related problems and other technical considerations. The most obvious asset of such an organization is the tremendous speed it offers for retrieval. Suitable programs can also be developed to make the organization extremely flexible. In addition, we believe the macroscopic concept of logical design, away from scanning, from searching, from addressing, and from counting, is equally important. We must, at all cost, free ourselves from the burdens of detailed local problems which only befit a machine low on the evolutionary scale of machines.

On the other hand, the emphasis on distributed logic introduces a number of physical problems. If a cell memory is to be practically useful, it must have many thousands, or perhaps millions of cells. Each cell must therefore be made of physical components which are less than miniature in size, and which must each consume extremely tiny amounts of power. Furthermore, because the cells are all identical, mass production techniques should be developed in which a whole block of circuitry can be formed at once.

Because the coordination of vast amounts of information is essential to scientific, economic, and military progress, the type of organization exemplified by the cell memory needs to be explored and explored extensively. The research on machine organization, however, cannot stand alone; the success of this research will depend also on the success in other fields of research: microminiaturization, integrated logic, and hyper-reliable circuit design.

ACKNOWLEDGEMENT

The writer is indebted to Messrs. S. H. Washburn, C. A. Lovell, and W. Keister for their support in this work. The writer also wishes to acknowledge the benefit he received from many discussions with Mr. M. C. Paull and with his other colleagues.

REFERENCES

1. Albert E. Slade, The Woven Cryotron Memory, Proc. Int. Symp. on the Theory of Switching, Harvard Univ. Press, 1959, p. 326.
2. R. R. Seeber and A. B. Linquist, Associative Memory with Ordered Retrieval, IBM Jour. of Res. and Dev., 5, 1962, p. 126.
3. R. S. Barton, A New Approach to the Functional Design of a Digital Computer, Proc. of the Western Joint Computer Conference, May 9 to 11, 1961, p. 393.
4. S. H. Unger, A New Type of Computer Oriented Towards Spatial Problems, Proc. of the IRE, 46, 1958, p. 1744.
5. P. M. Davis, A Superconductive Associative Memory, Proc. Spring Joint Computer Conference, May 1 to 3, 1962, p. 79.
6. V. L. Newhouse and R. E. Fruin, A Cryogenic Data Address Memory, Proc. Spring Joint Computer Conference, May 1 to 3, 1962, p. 89.
7. J. W. Crichton and J. H. Holland, A New Method of Simulating the Central Nervous System Using an Automatic Digital Computer, Tech. Report, Univ. of Mich., March, 1959.
8. H. Blum, An Associative Machine for Dealing with the Visual Field and Some of its Biological Implications, Tech. Report, Air Force Cambridge Research Labs., February, 1962.
9. R. F. Rosin, An Organization of an Associative Cryogenic Computer, Proc. Spring Joint Computer Conf., May 1 to 3, 1962, p. 203.
10. R. J. Segal and H. P. Guerber, Four Advanced Computers - Key to Air Force Digital Data Comm. Syst., Proc. Eastern Joint Computer Conference, December, 1961, p. 264.